Advant® OCS
with Master software

PC Elements
Advant® Controller 100 Series

Reference Manual

3BSE 000 504R301

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

ABB Master is a registered trademark of ABB Asea Brown Boveri Ltd. Switzerland.

Advant is a registered trademark of ABB Asea Brown Boveri Ltd. Switzerland.

MODBUS is a registered trademark of Gould Electronics Inc.

Siemens and all Siemens-products mentioned in this publication are trademarks of Siemens AG.

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Industrietechnik GmbH. ABB Industrietechnik GmbH assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Industrietechnik GmbH be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Industrietechnik GmbH be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Industrietechnik GmbH’s written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Copyright © ABB Industrietechnik GmbH 1999

3BSE001264/E

Template: 3BSE001286/E

TABLE OF CONTENTS

Introduction

PC Language 1
Call 1
Call Name 1
Call Parameters 2
Connection of PC Element Inputs and Outputs 2
Overflow Real Numbers 3
Element Libraries for MasterPiece Products 3
Programming with the PC Language 3
The Data Sheets 3
PC Elements According to Type 5
Short Form Description 9
Product Grouping 13
Execution Times 14
Space Requirements 18
Space Requirements for Inline Coded PC Elements 22

Data Sheets

ABS 23
ADD 24
ADD-MR 25
AND/OR 26
AND-O 28
AND-TRG 30
BGET 31
BLOCK 32
BSET 33
COM-CVI1 34
COM-CVO1 37
COMP-B 40
COMP-I 41
COMP-R 42
CON-PU1 44
CONTRM 50
CONV 53
CONV-BI 54
CONV-IB 58
COUNT 61
DER 63
DIV 66
DIV-MR 67

CONTENTS (continued)

FIFO 68
FILT-1P 70
FUNCM 73
FUNG-1V 74
INT 76
INV 79
LIM-N 80
MASTER 83
MAX 86
MFB-IN 88
MFB-OUT 93
MIN 96
MONO 98
MOVE 100
MUL 101
MULDIV 102
MUX-I 103
MUX-MI 105
MUX-MN 107
OR-A 109
OR-TRG 112
OSC-B 113
P-1 114
PCPGM 116
PDP 118
PI 121
PID 125
P_STEP 129
PIP 132
RAMP 135
REG 138
REG-G 140
SEQ 143
SHIFT 148
SLAVEM 150
SQRT 151
SR 152
SR-AA, SR-AO, SR-OO, SR-OA 153
SR-D 154
STEP 155
SUB 161

CONTENTS (continued)

SW.......... 162
SW–C.......... 163
SYS-CRC.......... 164
SYSL.......... 165
SYSTIME.......... 166
THRESH-L.......... 167
TIMER.......... 168
TON/ TOFF.......... 171
TON-RET.......... 173
TRIGG.......... 175
XOR.......... 176

Introduction

Advant Controller 100 Series is a range of products for control of various industrial processes. The system provides the ABB Master Programming Language methods, AMPL. AMPL is based on software implemented function blocks, called PC Elements, which are connected to each other into programs which form a complete control function. AMPL is used in all Advant Controllers, MasterPiece 200/1 and MasterPiece 90.

PC Language

The PC element is the smallest unit of the AMPL. Each element performs a complete function, such as a logical AND gate, a timer, or a feedback control element.

Call

Each element has its own call name that enables it to be accessed regardless of the computer system the element is in. Many of the elements have call parameters which allow size, function, data type, and other properties to be determined.

Call Name

The mnemonic call name provides information about the basic function of the element. Further information about the function of the element or the data types for which the element is intended is given by a suffix to the name. E.g. COMP-I, comparator for integers, and COMP-R, comparator for real numbers, see the table below.

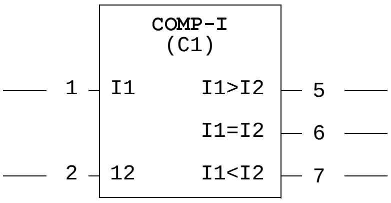
Suffix	Description
B	Boolean
I	Integer
R	Real number
T	Time
G	Group data
M	Memory function Multiplier in compound elements
L	Load function, Limiting function, Threshold function
RW	Read/Write
N	1-of-N addressing
D	Data
P	Number of poles
V	Variable
O	OR function in compound elements
A	AND function in compound elements

Call Parameters

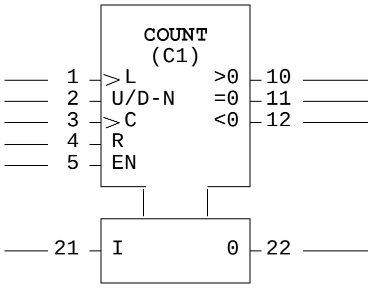
For structure elements the call parameters determine cyclicity, place in the cycle time table (determines the order of execution for PC modules having the same cyclicity) and, for a slave element, the identity of its master header. For a function element, the call parameters may specify the size, data type, number of inputs and outputs, etc.

Graphic Symbol

Each PC element has a graphic symbol similar to the one in the figure below.,



Normally, the call name of the element is written in the uppermost part of the symbol.
For certain functions, e.g. AND and OR, the name is replaced by a symbol. When the element has supervisory control signals, the terminals for these signals are collected in a separate part of the symbol according to the figure below.



Connection of PC Element Inputs and Outputs

Only inputs and outputs with the same data type can be interconnected. If, for example, an output with data of the type I is to be connected to an input of the type R, a code conversion element, e.g. CONV (I, R) must be inserted in the data path.

Designation	**Data Type**
I	Integer, 16 bit
IL	Integer, 32 bit
R	Real number
B	Boolean
T	Time
TR	Time, real representation
G	Group data, i.e. a collection of data with one of the above data types
A	Array

Error Handling, Integer Overflow

The range of allowable integer values is as follows.

For I= –32768 … 32767

For IL= –2147483648 … 2147483647

When the range is exceeded the output is set to the respective limiting value.

Overflow Real Numbers

The range of allowable real numbers is 9.2 x 10^{18} … –0.1 x 10^{18}. When the range is exceeded the output is set to the respective limiting value. The smallest magnitude that can be represented by a real number is 5.0 x 10^{-20}. Any output with an absolute value smaller than this is set to 0.0.

Element Libraries for MasterPiece Products

Each standard system has an element library which contains a defined subset of the total number of available PC elements. The Technical Description contains the definition of the PC library.

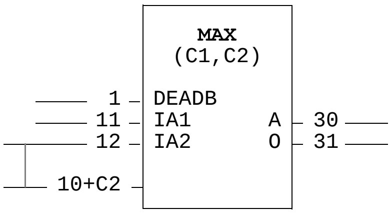
Programming with the PC Language

When programming a PC based control system, one chooses PC elements from the library, and includes them in a draft PC diagram. The elements are then interconnected on the diagram to define the exchange of data between them.

The information on the draft PC program is used when entering the PC program in source code form on a system aid. The aid translates the program into object code, which is loaded into the target system.

The Data Sheets

An element description starts with a short **Summary** that describes the main function of the element. The graphic symbol of the element is shown to the right of the summary. Symbols with a variable number of connections are shown in the figure below.

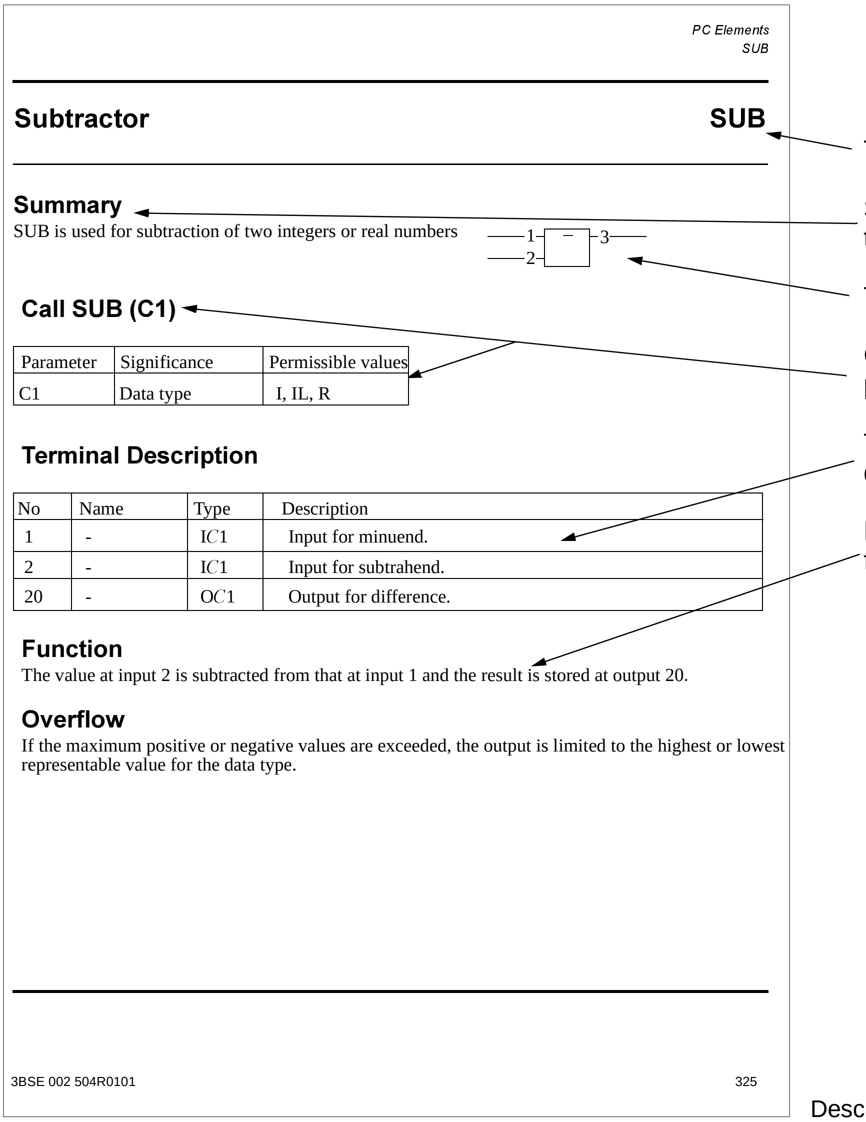


The calling of the element is described under the heading **Call.**

ELEMENT (C1, C2, … Cn)

If the element has call parameters, these are described in a table where their permissible values are stated. Element terminals are described under the heading **Connections**. This section provides a table giving each terminal number, name, type and application.

The type of connection is given as xy, where x is I or O if it is an input or an output terminal, Y denotes the data type previously described in the table below the heading Connections of PC Elements. The descriptions finish with a detailed functional description under the heading **Function**.



PC Elements
SUB

Subtractor SUB

Summary

SUB is used for subtraction of two integers or real numbers

Call SUB (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	-	IC1	Input for minuend.
2	-	IC1	Input for subtrahend.
20	-	OC1	Output for difference.

Function

The value at input 2 is subtracted from that at input 1 and the result is stored at output 20.

Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

3BSE 002 504R0101 325

The call name of the element.

Short description of the function of the element.

The graphic symbol of the element.

Call, with table over the call parameters, if any.

The terminals of the element are described in a table.

Detailed description of the elements function.

Description, schematic outline

PC Elements According to Type

Element Type	Element Name
Structure Elements	
Block Header	BLOCK
Control Module Header	CONTRM
Function Header	FUNCM
Master Header	MASTER
Program Header	PCPGM
Sequence Header	SEQ
Slave Header	SLAVEM
Step Header	STEP
Combinational Elements	
And Gate	AND
Inverter	INV
Or Gate	OR
Threshold Function	THRESH-L
Exclusive Or Gate	XOR
AND with OR gates at the inputs	AND-O
OR with AND gates at the inputs	OR-A
AND with trigger on output	AND-TRG
OR with trigger on output	OR-TRG
Bitwise Operation on the Integer Data Type	
Set one bit	BSET
Read one bit	BGET
Memory Elements	
Set/ Reset Memory	SR
Set/ Reset Memory with Data Input	SR-D
Set/ Reset Memory with And Inputs	SR-AA
Set/ Reset Memory with And/ Or Inputs	SR-AO
Set/ Reset Memory with Or/ And Inputs	SR-OA
Set/ Reset Memory with Or Inputs	SR-OO

Element Type	Element Name
Time Elements	
Monostable	MONO
Oscillator	OSC-B
Time Selector	TIMER
Time Delay Off	TOFF
Time Delay On	TON
Time Delay On with Retention	TON-RET
Trigger	TRIGG
Register Elements	
Queue Register	FIFO
Data Copying	MOVE
Register	REG
Group Data Register	REG-G
Shift Register	SHIFT
Switch Elements	
Switch	SW
Changeover Switch	SW-C
Multiplexers	
Multiplexer with Integer Address	MUX-I
Multiplexer with Memory and Integer Address	MUX-MI
Multiplexer with Memory and 1 of N Address	MUX-MN
Code Converter Elements	
Code Converter	CONV
Code Converter, Boolean to Integer	CONV-BI
Code Converter, Integer to Boolean	CONV-IB
Arithmetic Elements	
Absolute Value	ABS
Adder	ADD
Divider	DIV
Limiter	LIM-N
Multiplier	MUL
Square Root	SQRT
Subtractor	SUB

Element Type	Element Name
Adder with Multiplied Output	ADD-MR
Divider with Multiplied Inputs	DIV-MR
Integer Scaling Element	MULDIV
Counter Elements	
Counter	COUNT
Comparator Elements	
Comparator for Boolean Values	COMP-B
Comparator for Integer	COMP-I
Comparator for Real Numbers	COMP-R
Maximum Selector	MAX
Minimum Selector	MIN
Function Generators	
Function Generator with 1 Variable	FUNG-1V
Feedback and Filter Elements	
Three State Controller	CON-PU1
Derivator	DER
Filter with 1 Pole	FILT-1P
Integrator	INT
P Function Proportional	P-1
PDP Function	PDP
PI-Regulator	PI
PID Algorithm	PID
PIP-Function	PIP
Ramp Generator	RAMP
MasterFieldbus Communication Elements	
Data from Convertor	COM-CVI1
Data to Convertor	COM-CVO1
Data from Advant Controller 110	MFB-IN
Data to Advant Controller 110	MFB-OUT
Supervision Elements	
Calculation of Checksums	SYS-CRC

Element Type	Element Name
Calculation of System Load	SYSL
Element for Actual Time and Date	SYSTIME

Short Form Description

Element	Description
ABS	Calculates the absolute value of an integer or a real number. The result is multiplied by a scale factor.
ADD	Adds up to 19 integers or real numbers.
ADD-MR	Adds two groups of up to 19 real numbers each and subtracts the second from the first. The result is multiplied by a real number.
AND	Gives the Boolean product of up to 19 Boolean values.
AND-O	Gives the Boolean product of up to 9 Boolean values and up to 5 Boolean sums of a maximum of 9 Boolean values each.
AND-TRG	Used when the output of an AND gate should be triggered.
BGET	Reads the value of the selected bit from an integer or integer long value
BLOCK	Module header for blocks.
BSET	Sets the value of the selected bit in an integer or integer long value
COM-CVI1	Used as a communication link between a PC program and a converter TYRAK-L. Fetches information from the convertor.
COM-CVO1	Used as a communication link between a PC program and a converter TYRAK-L. Gives set points to the converter.
COMP-I	Comparator for Integer
COMP-B	Used to compare two bits.
COMP-R	Used to compare two real numbers.
CON-PU1	Three state controller with outputs for increase/decrease for control via an integrating actuator.
CONTRM	Module header for control modules.
CONV	Conversion of data between the types I, IL, R, T and TR.
CONV-BI	Converts data in BC, BCD, 1-of-N code, Gray code, and 16/32 bit packing, given at a number of Boolean inputs into data in integer form (I or IL).
CONV-IB	Converts data in integer form (I or IL) into data in BC, BCD, 1-of-N code and uncoded unpacking at a number of Boolean inputs.
COUNT	Pre-settable counter for counting pulses, up or down. With checking of the value relative to 0.
DER	Used to give derivation effect. The element has inputs and outputs for limit values, setting of gain, filter function for restricting the derivation effect and the possibility of following and external reference.
DIV	Used for dividing two real numbers or two integers. For integers the quotient is obtained along with a remainder at a special output.
DIV-MR	Used to divide one product of real numbers with another.
FIFO	Queue register with up to 9 queues each with a maximum of 64 places.

Element	Description
FILT-1P	1-pole low-pass filter with options for limiting the output signal and for following an external reference.
FUNCM	Module header for function module.
FUNG-1V	Generates a piece-wise linear function of one variable using tables, with up to 255 elements. If the function is strictly monotonous the inverse of the function can be generated by balancing.
INT	Used to give integration effect. The element has inputs and outputs for limit values, setting of gain and following with an external reference.
INV	Used to invert a Boolean value.
LIM-N	Used for limiting integers, real numbers or time values. Several limit values may be selected.
MASTER	Module header for master modules.
MAX	Selects the largest of up to 19 integers or real numbers.
MFB-IN	The element MFB-IN is used to transfer data to an Advant Controller 110 from an Advant Controller 410/ 450 connected via MasterFieldbus.
MFB-OUT	The element MFB-OUT is used to transfer data from an Advant Controller 110 to an Advant Controller 410/ 450 connected via MasterFieldbus.
MIN	Selects the smallest of up to 19 integers or real numbers.
MONO	Used to generate a pulse with a given duration when the input changes from 0 to 1. The pulse duration is specified at an input. Retriggability is controlled with an input.
MOVE	Used to move data, e.g. from one place in the database to another.
MUL	Used for multiplication of up to 19 integers or real numbers.
MULDIV	Scaling element for integer data
MUX-I	Selector for up to 19 inputs. Integer address.
MUX-MI	Selector for up to 19 inputs. Integer address and memory.
MUX-MN	Selector for up to 19 inputs. 1-of-N address and memory.
OR	Gives the Boolean sum of up to 19 Boolean values.
OR-A	Gives the Boolean sum of up to 9 Boolean values and up to 5 Boolean products each of a maximum of 9 Boolean values.
OR-TRG	Used when the output of an OR gate should be triggered.
OSC-B	Oscillator for a Boolean variable with variable frequency and pulse time.
P-1	Used as an amplifier in feedback systems. The element permits output signal limiting and following.
PCPGM	Module header for program.
PDP	Used to give proportional effect and limited derivation effect. The element permits limiting of the output signal and following.

Element	Description
PI	Used as a standard PI regulator for serial compensation in feedback systems. The element permits limiting of the output signal and following.
PID	PID controller used to create simple control loops on the function group and single-variable level.
PIP	Used to give proportional effect and limited integration effect. The element permits limiting of the output signal and following.
RAMP	Used to limit the rate of change of a signal. The element permits limiting of the output signal and following.
REG	Used as a memory element with an optional number of positions.
REG-G	Used to assemble simple variables into one single variable of group type.
SEQ	Module header for a sequence.
SHIFT	Used as a shift register with optional length.
SLAVEM	Module header for a slave. Subordinate to a MASTER.
SQRT	Calculates the square root of a real number. The result is multiplied by a scale factor.
SR	Memory for Boolean variables. The element has one Set and one Reset input.
SR-AA	Memory for Boolean variables. The element has And gates with up to 9 inputs on the Set and Reset inputs.
SR-AO	Memory for Boolean variables. The element has one And gate with up to 9 inputs on the Set input and one Or gate with up to 9 inputs on the Reset input.
SR-D	Memory for Boolean variables. The element has one Set and one Reset input and data and clock inputs for clocking in data.
SR-OA	Memory for Boolean variables. The element has one Or gate with up to 9 inputs on the Set input and one And gate with up to 9 inputs on the Reset input.
SR-OO	Memory for Boolean variables. The element has Or gates with up to 9 inputs on the Set and Reset inputs.
STEP	Module header for a step. Subordinate to an SEQ.
SUB	Subtraction of one real or one integer number from another.
SW	Switching element with up to 9 closing channels.
SW-C	Switching elements with up to 9 change over channels.
SYS-CRC	Calculation of checksums
SYSL	Calculation of system load
SYSTIME	Actual time and date
THRESH-L	Used to determine when more than a given number of Boolean signals are set to 1.

Element	Description
TIMER	Used to generate pulses at up to 9 outputs. The pulse duration is determined for each output by to times within a day.
TOFF	Delay for turning a Boolean signal off.
TON	Delay for turning a Boolean signal on.
TON-RET	Delay for turning a Boolean signal on with accumulating time measurement.
TRIGG	Provides a pulse during one program cycle when the input signal changes from 0 to 1.
XOR	Used to give a signal when one and only one of the input signals is true.

Product Grouping

This page shows which elements are included in the different Advant Controller 100 Series software packages.

Advant Controller 110 with BASE SW

ABS, ADD, ADD-MR, AND, AND-O, AND-TRG

BLOCK, BGET, BSET

COM-CVI1, COM-CVO1, COMP-B, COMP-I, COMP-R, CONTRM, CONV, CONV-BI, CONV-IB, COUNT

DIV, DIV-MR

FIFO, FUNCM

INV

LIM-N

MASTER, MAX, MFB-IN, MFB-OUT, MIN, MONO, MOVE, MUL, MULDIV, MUX-I, MUX-MI, MUX-MN

OR, OR-AND, OR-TRG, OSC-B

PCPGM

REG

SEQ, SHIFT, SLAVEM, SQRT, SR, SR-AA, SR-AO, SR-D, SR-OA, SR-OO, STEP, SUB, SW, SW-C, SYS-CRC, SYSL, SYSTIME

TRESH-L, TIMER, TOFF, TON,

TON-RET, TRIGG,

XOR.

Advant Controller 110 with OPT 1

CON-PU1, DER, FILT-1P, FUNG-1V, INT, P-1, PDP, PI, PID, PIP, RAMP, REG-G.

Execution Times

Name	Execution Time, typical (µs) (1)	Notes
ABS	22 50 60	C1=I C1=IL C1=R
ADD	9+2×C2 9+3×C2 32×C2–11	C1=I C1=IL, C2≥2 C1=R
ADD-MR	15×(1+C1+C2)	
AND(2)	7+2.5×C1	(1.4+1.2×C1), ( )=”inline” time
AND-O	18+6× (C1+C2+C3+C4+C5+C6)	
AND-TRG	12+2,8×C1	
BGET	14	
BLOCK	8	
BSET	18	
COM-CVI1	740+630×C1+10×C2	
COM-CVO1	1350+290×C1	
COMP-B	10,8	
COMP-I	16	
COMP-R	55+8×(C1+C2)	
CON-PU1	31 301 31 501	C1=0 At PF×TS Not at PF×TS C1=1 At PF×TS Not at PF×TS
CONTRM	690	
CONV	43 38 <95	I,IL→ R R→I,IL all other conversions

Name	Execution Time, typical (μs) (1)	Notes
CONV-BI	31+4×C3	C1=I, C2=1 (BC→I)
	37+7×C3	C1=I, C2=2 (BCD→I)
	29+2×C3	C1=I, C2=3 (1 of N→I)
	31+7×C3	C1=I, C2=4 (Gray→I)
	21+8×C3	C1=I, C2=5 (16/32BC→I)
	31+4×C3	C1=IL, C2=1 (BC→IL)
	37+7×C3	C1=IL, C2=2 (BCD→IL)
	29+2×C3	C1=IL, C2=3 (1 of N→IL)
	31+7×C3	C1=IL, C2=4 (Gray→IL)
	21+8×C3	C1=IL, C2=5 (16/32BC→IL)
CONV-IB	37+4×C3	C1=I, C2=1 (I→BC)
	78+4×C3	C1=I, C2=2 (I→BCD)
	36+4×C3	C1=I, C2=3 (I→1 of N)
	32+5×C3	C1=IL, C2=1 (IL→BC)
	78+5×C3	C1=IL, C2=2 (IL→BCD)
	36+4×C3	C1=IL, C2=3 (IL→1 of N)
COUNT	19	
DER	180	
DIV	25	C1=I
	210	C1=IL
	75	C1=R
DIV-MR	55+32×(C1+C2)	
FIFO	39+3×C2	C1=I, B
		C1=IL, R, T, TR
FILT-1P	220	
FUNCM	3.125	
FUNG-1V	250	
INT	130	
INV	7	
LIM-N	37+2×C2	
MASTER	690	
MAX	14+5×C2	C1=I
	14+7×C2	C1=IL
	12+9×C2	C1=R

Name	Execution Time, typical (µs) (1)	Notes
MFB-IN	315+(C1+C2+C3+C4+C5+C6)×20	
MFB-OUT	360+(C1+C2+C3+C4+C5+C6)×3	
MIN	14+5×C2 14+7×C2 12+9×C2	C1=I C1=IL C1=R
MONO	23	
MOVE	8+4×C2	
MUL	7+6×C2 29×C2–14 39×C2–22	C1=I C1=IL, C2≥2 C1=R
MULDIV	18,8	
MUX-I	18	
MUX-MI	28+2.5×C2	
MUX-MN	28+2.5×C2	
OR(2)	4+2.5×C1	(1.4+1.2×C1), ( )=”inline” time
OR-A	18+6× (C1+C2+C3+C4+C5+C6)	
OR-TRG	12+2,8×C1	
OSC-B	17	
P-1	140	
PCPGM	690	
PDP	260	
PI	240	
PID	1500	
PIP	270	
RAMP	220	
REG	16+3.5×C2	
REG-G	20+3×C2+10×C3+2×C4	
SEQ	1050 -at normal STEP execution 1400 -at STEP-CHANGE	C3=C4=C5=0 Including STEP
SHIFT	12	
SLAVEM	3.125	

Name	**Execution Time, typical (μs)** (1)	**Notes**
SQRT	170	
SR(2)	9	(4.4)
SR-AA	16+3×(C1+C2)	
SR-AO	16+3×(C1+C2)	
SR-D	14	
SR-OA	16+3×(C1+C2)	
SR-OO	16+3×(C1+C2)	
STEP		See SEQ
SUB	11 14 49	C1=I C1=IL C1=R
SW	18+6×C2	
SW-C	14+8×C2	
SYS-CRC	40	
SYSL	40+80×C1	
SYSTIME	210	
THRESH-L	22+2×C2	
TIMER	30+16×C1	
TOFF	10	
TON	10	
TON-RET	14	
TRIGG	11	
XOR	10	(3.5), ( )=inline time
Inline begin + end	5.2	onetime overhead for an inline sequence!
Type Circuits begin + end	6,25	onetime overhead for a type circuit

(1) execution times refer to a PM632 CPU. At PM633, PM634 CPUs typical times will be half of that.

(2) these elements are available as Inline code (see “Space Requirements for Inline Coded PC Elements)

Space Requirements

The following pages show the memory space requirements for PC elements when all the inputs of each element are connected to the output of another element.

Name	PC Statement (byte)	Local Data Area (byte)	Notes
ABS	14	2 4	C1=I C1=IL
ADD	10+2×C2	2 4	C1=I C1=IL, R
ADD-MR	12+2×C1+2×C2		
AND	10+2×C1	2	
AND-O	14+2×(C1+C2+C3+C4+C5+C6)	2	
AND-TRG	12+2×C1	4	
BGET	14		
BLOCK	12	2	
BSET	18		
COM-CVI1	2+16×2+46×C1+4×C2	44+C1×48+C2	
COM-CVO1	48+34×C1	64	
COMP-B	12	2	
COMP-I	18	6	
COMP-R	18+4×(C1+C2)	8+2×(C1+C2)	
CON-PU1	54	90	
CONTRM	38	4+14	
CONV	14	4	C1=I C1=IL, R, T, TR
CONV-BI	24+2×C3	6 8	C1=I C1=IL
CONV-IB	24+2×C3	12+2×C3	
COUNT	30	10 12	C1=I C1=IL
DER	34	44	
DIV	18	4 8	C1=I C1=IL

Name	PC Statement (byte)	Local Data Area (byte)	Notes
DIV-MR	10+2(C1+C2)	2	
FIFO	24+4×C2	16+2×(C2+C2×C3) 16+4×(C2+C2×C3)	C1=I, B C1=IL, TR, R, T
FILT-1P	30	40	
FUNCM	6	0	
FUNG-1V	26	36	
INT	32	32	
INV	10	2	
LIM-N	24+6×C2	14 20	C1=I C1=IL, R, T, TR
MASTER	38	4+14	
MAX	14+2×C2	4 6	C1=I C1=IL
MFB-IN	22+2(C1+C2+C3+C4+C5+C6)	8+4*(C1+C2+C3+C4+C5)+2*C6	
MFB-OUT	22+2(C1+C2+C3+C4+C5+C6)	8	
MIN	14+2×C2	4 6	C1=I C1=IL
MONO	18	12	
MOVE	8+4×C2	2×C2 4×C2	C1=B, I C1=IL, R, T, TR
MUL	10+2×C2	2 4	C1=I C1=IL, R
MULDIV	16	4	
MUX-I	14+2×C2	4 6	C1=B, I C1=IL, R, T, TR
MUX-MI	22+2×C2	6 8	C1=B, I C1=IL, R, T, TR
MUX-MN	20+4×C2	6 8	C1=B, I C1=IL, R, T, TR
OR	10+2×C1	2	
OR-A	14+2×(C1+C2+C3+C4+C5+C6)	2	
OR-TRG	12+2×C1	4	

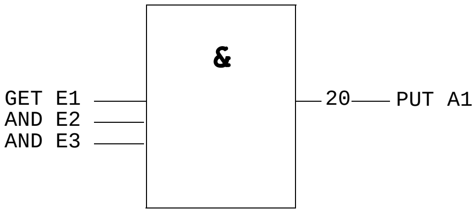
Name	PC Statement (byte)	Local Data Area (byte)	Notes
OSC-B	16	12	
P-1	28	14	
PCPGM	26	2+14	
PDP	32	48	
PI	38	36	
PID	52	90	
PIP	32	44	
RAMP	34	52	
REG	16+4×C2	2+2×C2 2+4×C2	C1=B, I C1=IL, R, T, TR
REG-G	28+2×C2+2×C3	6+2×C5 6+4×C5	C1=B, I C1=IL, R, T, TR
SEQ	20+14×C3+8×C4+4×K (K = number of STEP's)	200+14+2×C3+8×C4	
SHIFT	24	2+2×C2 2+4×C2	C1=B, I C1=IL, R, T, TR
SLAVEM	6	–	
SQRT	14	6	
SR	12	2	
SR-AA	10+2×(C1+C2+C3+C4+C5+C6)		
SR-AO	10+2×(C1+C2+C3+C4+C5+C6)		
SR-D	18	4	
SR-OA	10+2×(C1+C2+C3+C4+C5+C6)		
SR-OO	10+2×(C1+C2+C3+C4+C5+C6)		
STEP	22+8×C1+4×C2+4×C3+2×C4	2+6×C1	
SUB	14	2 4	C1=I C1=IL, R
SW	10+6×C2	2 × C2 4 × C2	C1=B, I C1=IL, R, T, TR
SW-C	12+2×C2	2 × C2 4 × C2	C1=B, I C1=IL, R, T, TR

Name	PC Statement (byte)	Local Data Area (byte)	Notes
SYS-CRC	14	8	
SYSL	10+8×C1	8+6×C1	
SYSTIME	22	16	
THRESH-L	10+2 ×C2	2	
TIMER	12+12×C1	12+2 × C1	
TOFF	16	12	
TON	16	12	
TON-RET	18	12	
TRIGG	12	4	
XOR	12	2	
Type Circuit begin + end	8+4		onetime overhead for a type circuit

Space Requirements for Inline Coded PC Elements

Name	PC Statement (byte per terminal)	Local Data Area (byte)	Notes
AND	6	2	
OR	6	2	
SR	10	2	
XOR	8	2	

.



Example for assignment of inline operations to PC element terminals

Generation of Inline Code

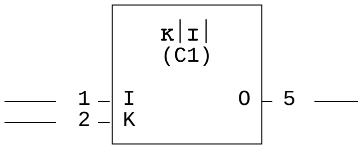
Inline code is automatically generated by the Advant Station 120 Engineering Station if at least two PC elements of types AND, OR, SR, or XOR follow another order. From the user’s viewpoint there is no difference between a normal PC element and the corresponding inline element.

Inline PC elements are used for speed optimization of binary operations. INV can be an additional attribute of a GET-, AND-, OR-, SR-, or XOR-terminal.

Absolute Value **ABS**

Summary

ABS (**ABS**olute value) is used to form the absolute value of an integer or a real number. The absolute value can be multiplied with an optional value.



PC element AB

Call ABS (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	I	IC*1*	**I**nput. Input for number, the absolute value of which is to be obtained.
2	K	*OC1*	Input for multiplication factor K.
5	O	*OC1*	**O**utput. Output for the product of the value at input K and the absolute value of the value at the input I.

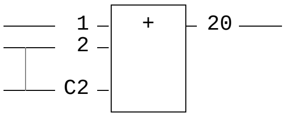
Function

The value at input K is multiplied by the absolute value of the input I and the result is stored at the output O.

Adder ADD

Summary

ADD is used for addition of an optional number of integers or real numbers



PC element ADD

Call ADD (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2..19

Terminal Description

No	Name	Type	Description
1	-	I*C1*	Input for addend.
2	-	I*C1*	Input for addend.
C	-	I*C1*	Input for addend.
20	-	I*C1*	Output for sum.

Function

The values at the inputs 1 … C2 are added and the sum is stored at the output 20.

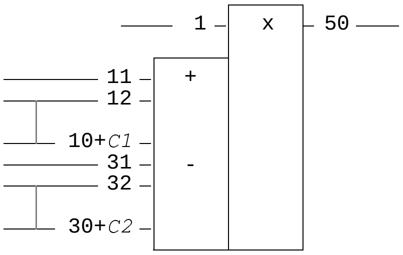
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Adder with Multiplied Output ADD-MR

Summary

ADD-MR (**ADD**er **-** with **M**ultiplier **R**eal numbers) is used for addition of an optional number of real numbers. The numbers are added in two groups, each of a maximum of 19 numbers, after which the second group is subtracted from the first. The result is multiplied with a real number.



PC element ADD-MR

Call ADD-MR (C1, C2)

Parameter	Significance	Permissible values
C1	Number of positive inputs in the addition element.	1 .. 19
C2	Number of negative inputs in the addition element.	1 .. 19

Terminal Description

No	Name	Type	Description
1	-	IR	Input for multiplication factor.
11	-	IR	Input for addend with positive weight factor.
12	-	IR	Input for addend with positive weight factor.
: 10+*C1*	-	IR	Input for addend with positive weight factor.
31	-	IR	Input for addend with negative weight factor.
32	-	IR	Input for addend with negative weight factor.
: 30+*C2*	-	IR	Input for addend with negative weight factor.
50	-	OR	Output for result.

Function

The sum of the real numbers at the inputs 31 .. 30 + *C2* is subtracted from the sum of the inputs 11 .. 10 + *C1*, after which the result is multiplied with the value at input 1. The result is stored at the output 50.

Overflow

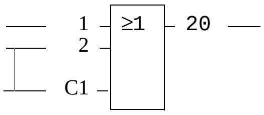
If the maximum positive or negative real value is exceeded, the output is limited to the greatest or lowest representable value respectively.

And Gate / Or Gate

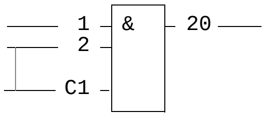
AND/OR

Summary

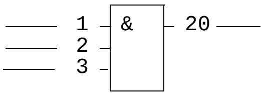
OR and AND elements are used to form general combinatorial expressions with Boolean variables. The elements can have a maximum of 19 inputs.



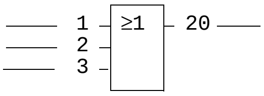
PC element OR



PC element AND



PC element AND (3)



PC element OR (3)

Call OR (C1) AND (C1)

Parameter	Significance	Permissible values
C1	Number of inputs	2..19

Terminal Description

No	Name	Type	Description
1	--	IB	Input
2	--	IB	Input
: C1	--	IB	Input
20	--	OB	Output

Function- OR element

The output signal is set if any of the inputs are set. See the table below.

Table OR

1	2	3	20
0	0	0	0
0	0	1	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1

Table OR

1	**2**	**3**	**20**
1	1	1	1

AND element

The output signal is set to 1 if all of the inputs are 1. See the table below.

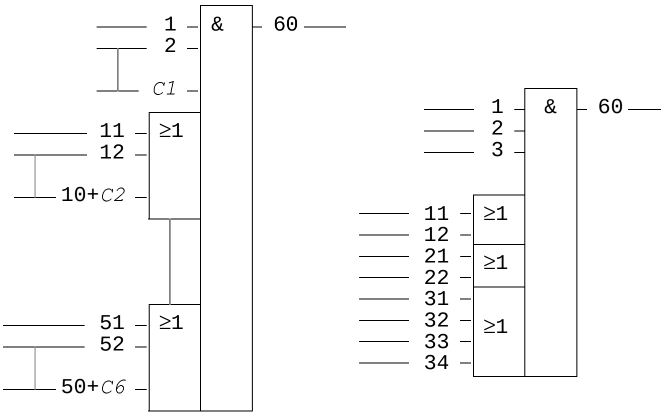
Table AND

1	**2**	**3**	**20**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

And Gate With Or Inputs AND-O

Summary

AND-O (**AND -** with **O**r gates on the inputs) is used to form general combinatorial expressions with Boolean variables.



PC element AND-

Call AND-O (C1, C2, C3…C*n*)

Parameter	Significance	Permissible values
C1	Number of inputs in the AND part of the element	0..9
C2	Number of inputs in the first OR gate of the element	0..9
C3	Number of inputs in the second OR gate of the element	0..9
: C*n*(1)	Number of inputs in the last OR gate of the element	0..9

(1) 1≤*n*≤6
1≤*C1*+*C2*+...+C*n*≤40

Terminal Description

No	Name	Type	Description
1	--	IB	Input direct to the AND gate in the element.
2	--	IB	Input direct to the AND gate in the element.
: *C1*	--	IB	Input direct to the AND gate in the element.
11	--	IB	Input to the first OR gate in the element.
12	--	IB	Input to the first OR gate in the element.
: 10+*C2*	--	IB	Input to the first OR gate in the element.
21	--	IB	Input to the second OR gate in the element.

Terminal Description

No	Name	Type	Description
22	--	IB	Input to the second OR gate in the element.
: 20+*C3*	--	IB	Input to the second OR gate in the element.
: 10x(*n*-1)+1	--	IB	Input to the last OR gate in the element.
10x(*n*-1)+2	--	IB	Input to the last OR gate in the element.
: 10x(*n*-1)+C*n*	--	IB	Input to the last OR gate in the element.
60	--	IB	Output

Function

More rapid execution than if separate elements are used can be obtained by using assembled elements such as AND-O. The inputs need then only be tested until the value of the output can be determined. If input 1 is 0, the output can be reset irrespective of the other inputs. See the table below.

Table AND-O (3, 3, 3)

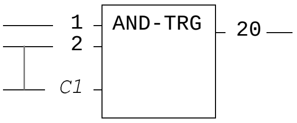
1	2	3	11	12	13	21	22	23	60
0	x(1)	x	x	x	x	x	x	x	0
1	0	x	x	x	x	x	x	x	0
1	0	0	x	x	x	x	x	x	0
1	1	1	0	0	0	x	x	x	0
1	1	1	0	0	1	0	0	0	0
1	1	1	0	0	1	0	0	1	1
1	1	1	0	0	1	0	1	x	1
1	1	1	0	0	1	1	x	x	1
1	1	1	0	1	x	0	0	0	0
1	1	1	0	1	x	0	0	1	1
1	1	1	0	1	x	0	1	x	1
1	1	1	0	1	x	1	x	x	1
1	1	1	1	x	x	0	0	0	0
1	1	1	1	x	x	0	0	1	1
1	1	1	1	x	x	0	1	x	1
1	1	1	1	x	x	1	x	x	1

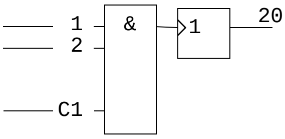
(1) x indicates that the input has no effect on the value of the output or the execution time.

And Gate With Trigger on Outputs AND-TRG

Summary

AND-TRG (**AND** - with **TR**i**G**ger on the output) is used when the output of an AND gate should be triggered.





Call AND-TRG (C1)

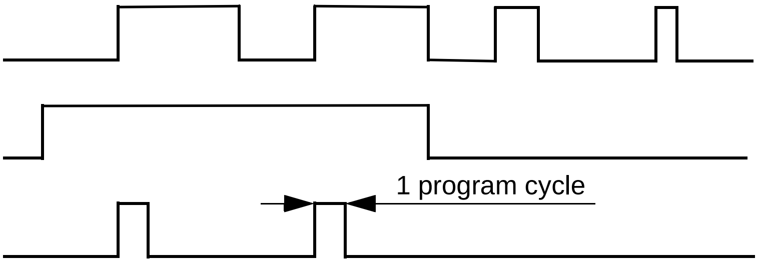
Parameter	Significance	Permissible values
C1	Number of AND inputs	2..19

Terminal Description

No	Name	Type	Description
1	--	IB	Input to the AND gate.
2	--	IB	Input to the AND gate.
: *C1*	--	IB	Input to the AND gate.
20	--	OB	Output

Function

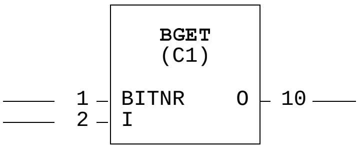
When the result of the AND gate is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the result of the AND gate. For the output signal to be set, the result of AND gate must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.



Read one bit BGET

Summary

BGET is used to read one selected bit from an integer or long integer value. The integer usually contains packed boolean data.



PC element BGE

Call BGET (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description	Values
1	BITNR	II	The selected **BIT N**umbe**R** to read	0..15 (C1=I) or 0..31 (C1=IL)
2	I	I*C1*	**I**nput. Data input	
10	O	OB	**O**utput. Status of the bit selected.	

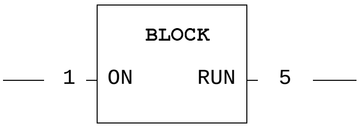
Function

The bit value of input I, specified by BITNR, is loaded to output O. If BITNR is not the range 0..15 (C1=I) or 0..31 (C1=IL), then output O is set to 0.

Block Header BLOCK

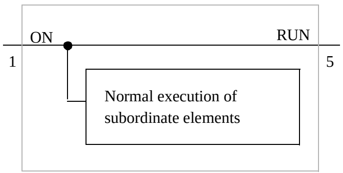
Summary

Block header BLOCK is used to connect and disconnect the execution of several PC elements.



PC element BLOCK

Call BLOCK



Function diagram

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) for normal execution
5	RUN	OB	**RUN** indicates execution of the elements in the block.

Function

BLOCK is used to permit the connection and disconnection of a number of PC elements in a control module, slave or sequence step. A block may not contain structure elements.

Execution

For normal execution of a block, the superior execution unit must be under normal execution, and the input ON of the block header must be set to 1. The execution of the block is then performed according to the conditions of the superior execution unit. If the ON input is set to 0 after the block has been executed, the calculated data remains until the next time the block is executed. If the superior execution unit is being executed in the reset mode, the elements in the block will also be executed in that mode, irrespective of the state of the ON input.

Blocking

Blocking requires the ON input to be set to 0. At reset the elements in the block will stop executing, but the latest value at each output will be read out to the database unchanged. To reset the outputs, the superior execution unit must be executed in the reset mode. The superior execution unit will then read out 0-values to the database. Blocking from a service aid is performed independently of the ON input of the block in accordance with the status of the execution unit which includes the block. Note that time delays under BLOCK, e.g. TOFF, will increase the delay time with the time of blocking.

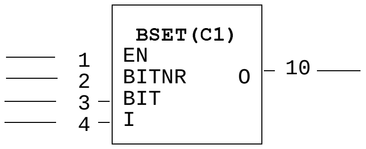
RUN

The output RUN is set only if normal execution is in progress, i.e. the superior execution unit is under normal execution and the ON input of the block is set.

Set one bit BSET

Summary

BSET is used to set one selected bit of an integer or long integer value.The integer usually contains packed boolean data.



PC element BSE

Call BSET (C1)

Parameter	**Significance**	**Permissible values**
C1	Data type	I, IL

Terminal Description

No	**Name**	**Type**	**Description**	**Values**
1	EN	IB	**EN**able. If EN is reset (to 0), then the current value I is loaded unchanged to output O.	
2	BITNR	II	Input for selecting the **BIT N**umbe**R** to set	0..15 (C1=I) or 0..31 (C1=IL)
3	BIT	IB	Input. Value of the **BIT**	0..1
4	I	I*C1*	**I**nput. Data input	
10	O	O*C1*	**O**utput. Status of the bit selected.	

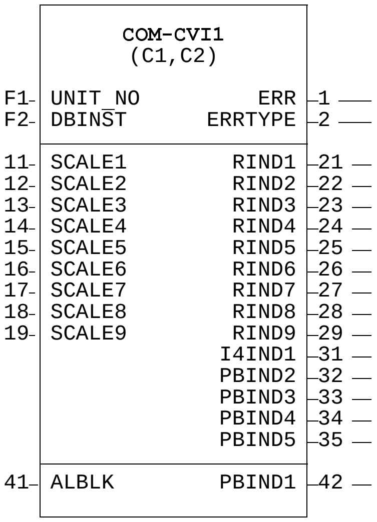
Function

If the value of input EN is set (to 1) then the data bit in the output value, determined by BITNR, is replaced by the value of the input bit BIT and the result is loaded to output O.

Data from Convertor COM-CVI1

Summary

The COM-CVI1 element is used as a communication link between a PC program in a controller and a convertor, TYRAK-L. The element is specially designed to fetch information from the convertor. For communication, a data base element of type CV unit is required.



PC element COM-CVI1

Call COM -CVI1(C1,C2)

Parameter	Significance	Permissible values
C1	Indicates whether arithmetic variables are to be included in the element. (Connection 11-35).	0..1
C2	Indicates whether MMC connection is to be included in the element. (Connection 41 and 42).	0..1

C1 =0<>C2=0

Terminal Description

No	Name	Type	Description
F1	UNIT_NO	FI	Number of the unit. Permitted values are 100-115, 200-215, 300-315, 400-415 i.e. 100 x Bus+Jumper address.
F2	DBINST	FI	Not used for Advant Controller 110 (Terminal exists for compatibility reasons with Advant Controller 400 series or MasterPiece 200/1).
1	ERR	OB	Error on initializing or outputs not updated.
2	ERRTYPE	OI	Error code
C1 dependent terminals:			
11	SCALE1	IR	Scale factor for output 21.
12	SCALE2	IR	Scale factor for output 22.

Terminal Description

No	Name	Type	Description
13	SCALE3	IR	Scale factor for output 23.
14	SCALE4	IR	Scale factor for output 24.
15	SCALE5	IR	Scale factor for output 25.
16	SCALE6	IR	Scale factor for output 26.
17	SCALE7	IR	Scale factor for output 27.
18	SCALE8	IR	Scale factor for output 28.
19	SCALE9	IR	Scale factor for output 29.
21	RIND1	OR	REAL 1 from convertor.
22	RIND2	OR	REAL 2 from convertor.
23	RIND3	OR	REAL 3 from convertor.
24	RIND4	OR	REAL 4 from convertor.
25	RIND5	OR	REAL 5 from convertor.
26	RIND6	OR	REAL 6 from convertor.
27	RIND7	OR	REAL 7 from convertor.
28	RIND8	OR	REAL 8 from convertor.
29	RIND9	OR	REAL 9 from convertor.
31	I4IND1	OIL	I x 4 from convertor (see ”Continuous data”).
32	PBIND2	OI	Block word for command acknowledgments.(1)
33	PBIND3	OI	Block word for status.(1)
34	PBIND4	OI	Block word for indication to MMC.(1)
35	PBIND5	OI	Block word for interlocking inputs etc.(1)
C2 dependent terminals:			
41	ALBLK	IB	Blocking of event analysis.
42	PBND1	OI	Block word for event handling.(1)

(1) PBIND1-5 each contain 16 logic signals. The content of these outputs is transparent to the element. The content can therefore be changed without changing the element.

Function, General

PC element COM-CVI1 is used to transfer variables (indications of measured values) to a higher level ABB Master system from a digital convertor (TYRAK-L). The MasterFieldbus is used for communication.

Arithmetical variables are transmitted on the bus as integers. Those outputs of the element that are of data type R (real) are scaled against the relevant scale factor (base value) before the value goes to the output. The element is divided into functions that can be chosen to suit the application.

Scaling

In Master, arithmetical variables are usually represented as real. Since the node works internally with integers (and fractions) scaling must be done in Master. This is done in the element for real outputs. The scale factor is an ordinary input and can be connected to a property in the database or specified as an operating parameter. The scale factor represents the highest positive number that can be represented as I x 2. Conversion from the integers of the node to a real value at the output is performed according to the following formula: output (R)=node value (I x 2) x (scale factor (R) / 32768)

To reduce the load on the element, only those inputs that are actually used are scaled. The element checks whether the scale factor is 0 and if it is, no scaling takes place and the corresponding output is set to 0. The quotient of the incoming scale factor and 32768 is calculated only every 20th cycle to reduce the load on the element. The result is buffered and used in other calculations. The scale factor should therefore not be dynamic.

Several PC Elements per Node

Since the element only reads data, it is permissible for the element to occur in several calls. This is useful when the event analysis subfunction is required to be placed under a short cycle time, while still executing the remaining functions during the normal cycle time.

Continuous Data

The element accesses DSCS 131 without semaphoring. This means that only two bytes of continuous data can be guaranteed. This limitation improves performance.

Database

The data base element CV unit is used to start and monitor communication. On starting, the element searches in the database for the unit that matches the bus and node specified by the function parameter UNIT_NO. The same database file is used for both order (COM-CVO1) and indication (COM-CVI1) elements.

Error Codes

The outputs ERR and ERRTYPE specify the error states that can arise. ERR is a logic output that is set to 1 when an error is detected in the transmission of data from the node. ERRTYPE mirrors the error state of the element but it continues to execute as usual. The table below shows the various types of error that are mirrored in ERRTYPE.

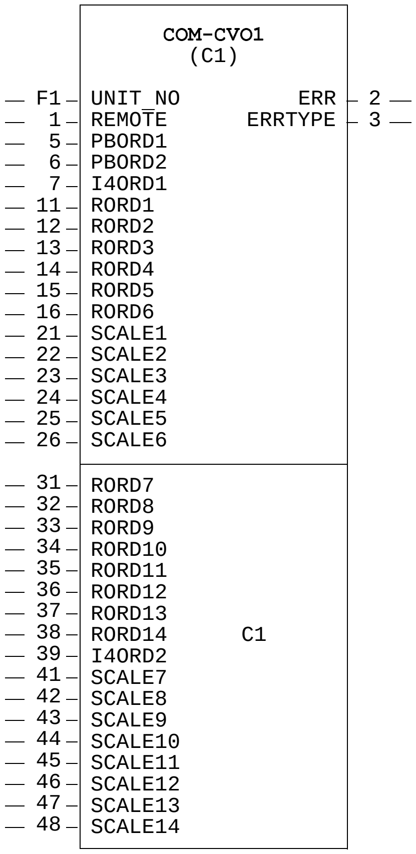
Error Types

Error Code	Description of error
–1	Communication with the convertor is not initiated.
–3	The data base element given by the parameter UNIT_NO is not accepted. This may be caused by an erroneous bus or node number.
0	OK
7	Connected node is not a convertor.
20	The received data is not valid
22	Communication board CI671 error.

Data to Convertor COM-CVO1

Summary

The COM-CVO1 element is used as a communication link between a PC program in a controller and a convertor, TYRAK-L. The element is specially designed to give set points to the convertor. For communication, a data base element of type CV unit is required.



PC element COM-CVC

Call COM-CVO1 (C1)

Parameter	Significance	Permissible values
C1	Indicates whether the input signals 31-48 are to be included in the element or not.	0..1

Terminal Description

No	Name	Type	Description
F1	UNIT_NO	FI	Number of the unit. Permitted values are 100-115, 200-215, 300-315, 400-415 i.e. 100 x Bus + Jumper address.
1	REMOTE	IB	Node ordered to remote mode.
2	ERR	OB	Error on initializing or communication error.

Terminal Description

No	Name	Type	Description
3	ERRTYPE	OI	Error code.
5	PBORD1	II	Block term 1(1).
6	PBORD2	II	Block term 2(1).
7	I4ORD1	IL	I x 4 to convertor.
11	RORD1	IR	REAL 1 to convertor.
12	RORD2	IR	REAL 2 to convertor.
13	RORD3	IR	REAL 3 to convertor.
14	RORD4	IR	REAL 4 to convertor.
15	RORD5	IR	REAL 5 to convertor.
16	RORD6	IR	REAL 6 to convertor.
21	SCALE1	IR	Scale factor for input 11.
22	SCALE2	IR	Scale factor for input 12.
23	SCALE3	IR	Scale factor for input 13.
24	SCALE4	IR	Scale factor for input 14.
25	SCALE5	IR	Scale factor for input 15.
26	SCALE6	IR	Scale factor for input 16.
If C1=1, the following option values will be transmitted (if C1=0, default values=0 are transmitted for these variables):			
31	RORD7	IR	REAL 7 to convertor.
32	RORD8	IR	REAL 8 to convertor.
33	RORD9	IR	REAL 9 to convertor.
34	RORD10	IR	REAL 10 to convertor.
35	RORD11	IR	REAL 11 to convertor.
36	RORD12	IR	REAL 12 to convertor.
37	RORD13	IR	REAL 13 to convertor.
38	RORD14	IR	REAL 14 to convertor.
39	I4ORD2	IIL	I x 4 to convertor.
41	SCALE7	IR	Scale factor for input 31.
42	SCALE8	IR	Scale factor for input 32.
43	SCALE9	IR	Scale factor for input 33.
44	SCALE10	IR	Scale factor for input 34.
45	SCALE11	IR	Scale factor for input 35.

Terminal Description

No	Name	Type	Description
46	SCALE12	IR	Scale factor for input 36.
47	SCALE13	IR	Scale factor for input 37.
48	SCALE14	IR	Scale factor for input 38.

(1) PBORD1-2 each contain 16 logical variables. The content is transparent to the element except bit 15 in PBORD1, which is reserved for the REMOTE input of the element. The meaning can therefore be changed without changing the element.

Function, General

PC element COM-CVO1 is used to transfer variables (orders) from a higher level system to a digital convertor (TYRAK-L). The MasterFieldbus is used for communication. The arithmetical input signals (R) of the element are rescaled and transmitted on the bus as integers (I x 2). For each MFB unit one pair of these two elements are built in each system.

Scaling

The inputs of the element of data type real (R) are rescaled to integers using a scale factor that is also of the data type real. This is done because integers are used internally in the node (TYRAK). The scale factor is an ordinary input and can be connected to a property in the database or specified as an operating parameter. The scale factor represents the highest positive number that can be represented as I x 2. Conversion from real input value to an integer is performed according to the following formula:

output (I x 2)=input (R) x 32768 / scale factor (R)

To reduce the load on the element, only those inputs that are actually used are scaled. The element checks whether the scale factor is 0 and if it is, no scaling takes place and the default value of the node is transmitted (=0). The quotient of 32768 and the incoming scale factor is calculated only every 20th cycle to reduce the load on the element. The result is buffered and used in other calculations. The scale factor should therefore not be dynamic. The element limits data to the node between an upper limit (32767) and a lower limit (–32768). The hexadecimal equivalent of the maximum value 32767 is $7FFF and of the minimum value -32768 it is $8000.

Data Base

The data base element CV unit is used to start and monitor communication. On starting, the element searches the database for the unit that matches the bus and node specified by the function parameter UNIT_NO.

Error Codes

The outputs ERR and ERRTYPE specify the error states that can arise. ERR is a logic output that is set to 1 when an error is detected in the transmission of data to the node. ERRTYPE mirrors the error state of the element but it continues to execute as usual. The table shows the various types of error that are mirrored in ERRTYPE.

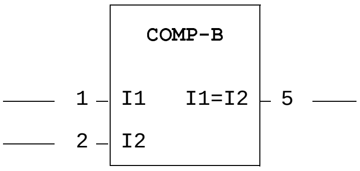
Error Code	Description of error
0	OK
-1	The data base element given by the PC element parameter UNIT_NO is missing.
-3	The data base element given by the parameter UNIT_NO is not accepted. This may be caused by an erroneous bus or node number.
22	Communication board CI671 error.

Comparator for Boolean values **COMP-B**

Summary

COMP-B (**COMP**arator **- B**oolean) is used to compare two integers.

Call COMP-B



PC element COMP-B

Terminal Description

No	Name	Type	Description
1	I1	IB	**I**nput **1**. Input, the value of which is to be compared with input I2.
2	I2	IB	**I**nput **2**. Input, the value of which is to be compared with input I1.
5	I1=I2	OB	**I**nput **1**=**I**nput **2**.

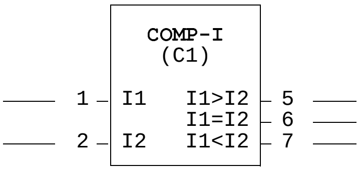
Function

The values at the two inputs are compared and the result of the comparison can be read at the output. The output signal is 0 if the input signals are different and 1 if they are equal.

Comparator for Integers COMP-I

Summary

COMP-I (**COMP**arator **- I**nteger) is used to compare two integers.



PC element COMP-I

Call COMP-I (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description
1	I1	IC1	**I**nput **1**. Input, the value of which is to be compared with input I2.
2	I2	IC1	**I**nput **2**. Input, the value of which is to be compared with input I1.
5	I1>I2	OB	**I**nput **1**>**I**nput **2**. Output which is set if the value at the input I1 is greater than the value at input I2.
6	I1=I2	OB	**I**nput **1**=**I**nput **2**. Output which is set if the value at the input I1 is equal to the value at input I2.
7	I1<I2	OB	**I**nput **1**<**I**nput **2**. Output which is set if the value at the input I1 is less than the value at input I2.

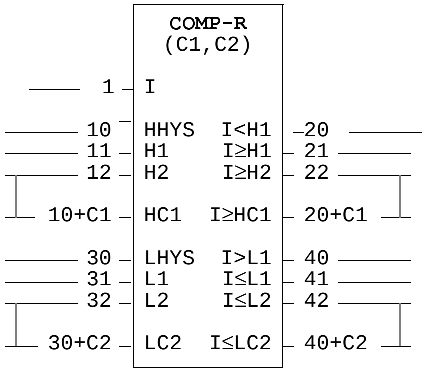
Function

The values at the two inputs are compared and the result of the comparison can be read at the outputs I1<I2, I1=I2 and I1>I2. At startup the output I1=I2 is set before the first execution independent of the values of the inputs I1 and I2.

Comparator for Real Numbers COMP-R

Summary

COMP-R (**COMP**arator **- R**eal) is used for limit value monitoring of real numbers with relation to several limits.



PC element COMP-R

Call COMP-R (C1, C2)

Parameter	Significance	Permissible values
C1	Number of limit values for high level	1..9
C2	Number of limit values for low level	1..9

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input, the value of which is to be compared with limit values H1..H*C1* and L1..L*C2*.
10	HHYS	IR	**H**igh **HYS**teresis. Hysteresis common for all H values. The hysteresis is specified as the difference between the I-value at which the H outputs are set and the value at which they are reset.
11	H1	IR	**H**igh **1**. Upper limit value 1.
12	H2	IR	**H**igh **2**. Upper limit value 2.
: 10+C1	HC1	IR	**H**igh ***C1***. Upper limit value C1.
20	I<H1	OB	**I**nput<**H1**. Output which is set if I is less than H1.
21	I≥H1	OB	**I**nput≥**H1**. Output which is set if I is greater than or equal to H1.
22	I≥H2	OB	**I**nput≥**H2**. Output which is set if I is greater than or equal to H2.
: 20+C1	I≥HC1	OB	**I**nput≥**H***C1*. Output which is set if I is greater than or equal to HC1.
30	LHYS	IR	**L**ow **HYS**teresis. Hysteresis common for all L values. The hysteresis is given as the difference between the I- value at which the L outputs are reset and the value at which they are set.
31	L1	IR	**L**ow **1**. Lower limit value 1.

Terminal Description

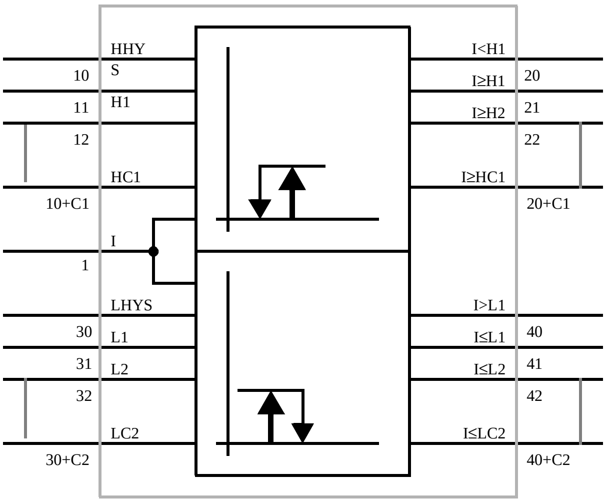
No	Name	Type	Description
32	L2	IR	**L**ow **2**. Lower limit value 2.
: 30+C2	LC2	IR	**L**ow ***C2***. Lower limit value C2.
40	I>L1	OB	**I**nput>**L1**. Output which is set if I is greater than L1.
41	I≤L1	OB	**I**nput≤**L1**. Output which is set if I is less than or equal to L1.
42	I≤L2	OB	**I**nput≤**L2**. Output which is set if I is less than or equal to L2.
: 40+C2	I≤LC2	OB	Input≤LC2. Output which is set if I is less than or equal to LC2.

Function

Input signal I is compared with the limit value specified at the inputs H1..H*C1* and L1..L*C2*. For upper limits, the output for the different limits will be set when the input I becomes equal to or greater than the limit value. For lower limits, the output is set when I becomes less than or equal to the limit value. Limits can be set optionally within the interval $-9.2\text{x}10^{18}$ … $9.2\text{x}10^{18}$. The lowest representable positive and negative values are $5.0\text{x}10^{-20}$ and $–5.0\text{x}10^{-20}$. The element put outputs 20 (I<H1) and 40 (I>L1) to 1 before the first execution independent of the input I.

Hysteresis

The input HHYS gives the hysteresis for all limits at high level and LHYS for all limits at low level. The hysteresis is given as the difference between the I values for which the different outputs are set and the values for which they are reset. At high level the outputs will be reset when I becomes lower than the limit minus the hysteresis HHYS. At low level the outputs will be reset when I becomes greater than the limit value plus the hysteresis LHYS.

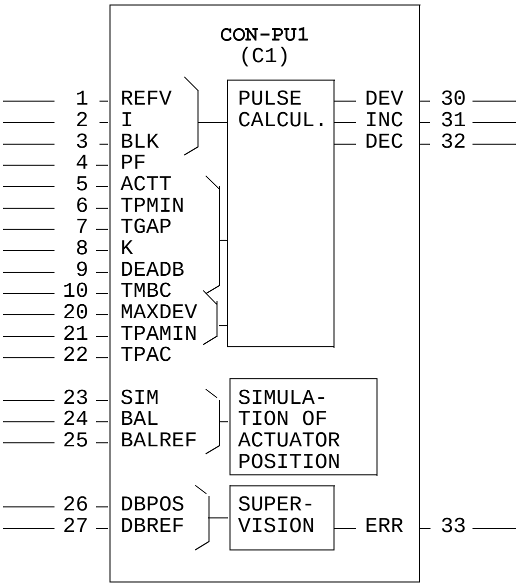


Function diagram

Three State Controller CON-PU1

Summary

CON-PU1 (**CON**troller **- PU**lsed **1**) is a three state controller intended to control, by means of pulses (increase/decrease), a process via an integrating actuator such as a shifting motor of a solenoid valve. CON-PU1 can be combined with, for example, a PI regulator for process control. If only pulse generation is desired, a simplified function can be chosen with a call parameter.



PC element CON-PU1

Call CON-PU1 (C1)

Parameter	Significance	Permissible values
C1	Selection of extended or ordinary function. The value 0 gives ordinary function.	0..1

Terminal Description

No	Name	Type	Description
1	REFV	IR	**REF**erence **V**alue. Input for setpoint.
2	I	IR	**I**nput value. Input for measured value.
3	BLK	IB	**BL**oc**K** pulse calculation.
4	PF	II	**P**ulse **F**requency Factor. Minimum pulse cycle time=PF x TS.
5	ACTT	ITR	**ACT**uator **T**ime. The delay time interval of the actuator.
6	TPMIN	ITR	**T**ime **P**ulse **MIN**imum. Min pulse length for excitation of the motor.
7	TGAP	ITR	**T**ime **GAP**. Compensation for play.
8	K	IR	Pulse length factor. Pulse=(KxDEVxACTT) /100+TGAP+TPMIN.

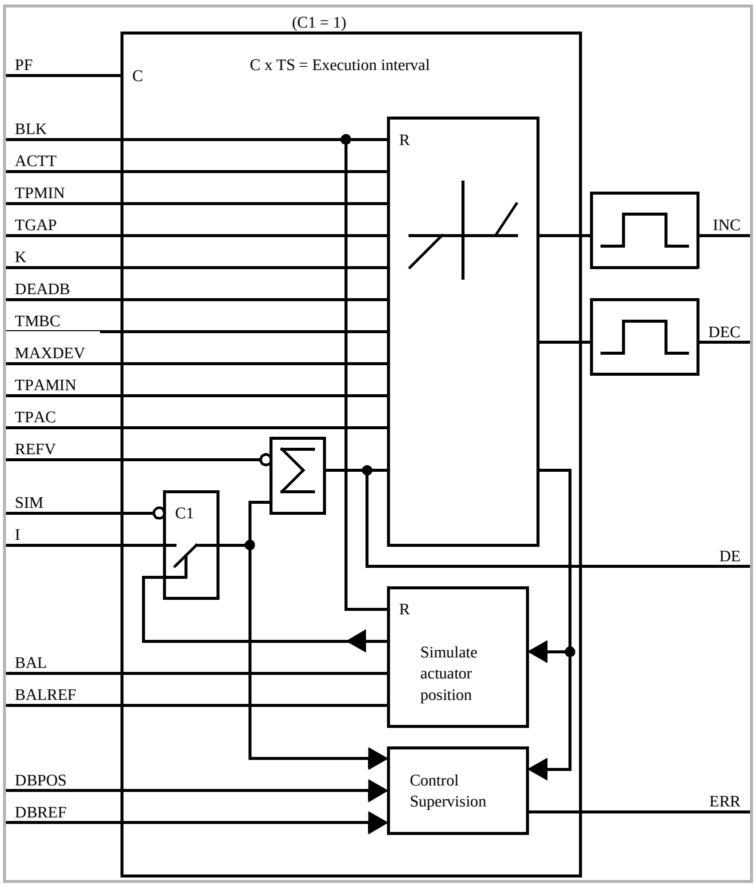
Terminal Description

No	Name	Type	Description
9	DEADB	IR	**DEADB**and. $\vert \text{DEV} \vert \geq$DEADB if a pulse is to be given.
10	TMBC	ITR	**T**ime **M**inimum **B**etween **C**hange. Minimum pause time after DEV having changed sign.
20(1)	MAXDEV	IR	**MAX**imum **DEV**iation. Maximum deviation for pulse length calculation. This value does not affect output DEV.
21(1)	TPAMIN	ITR	**T**ime **PA**use **MIN**imum. Minimum pause time.
22(1)	TPAC	IR	**T**ime **PA**use **C**onstant. Pause Time factor. Pulse time=TPAMIN+TPAC x Pulse duration.
23(1)	SIM	IB	**SIM**ulate actuator position. Input for activation of simulation of actuator position.
24(1)	BAL	IB	**BAL**ance. Input for activation of tracking.
25(1)	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when tracking.
26(1)	DBPOS	IR	**D**ead**B**and for **POS**ition change. Deadband for supervision of position change of actuator.
27(1)	DBREF	IR	**D**ead**B**and **REF**erence for position change alarm. When $\left\vert \int_{t0}^{tn} \text{DEVxK dt} \right\vert$>DBPOS a check is made to determine if $\vert I_{tn} - I_{t0} \vert$>DBREF and if so, the error signal ERR is set.
30	DEV	OR	Position **DEV**iation. DEV=I - REFV.
31	INC	OB	**INC**rease. Output for increase of actuator position.
32	DEC	OB	**DEC**rease. Output for decrease of actuator position.
33(1)	ERR	OB	**ERR**or. Actuator position has not followed the change commanded.

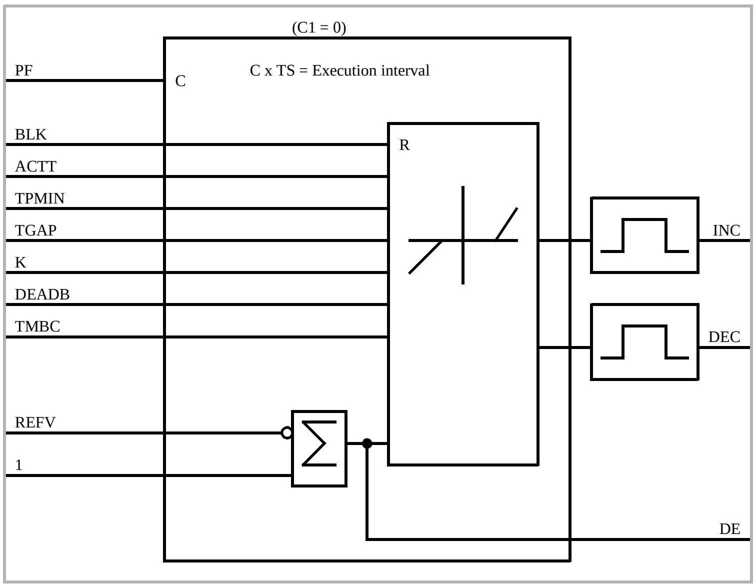
(1) Only used with extended function, (C1=1)

Function

The three state controller CON-PU1 is intended to control, by means of pulses at the outputs INC and DEC (increase/decrease respectively), a process via for example a shifting motor or a solenoid valve. With the call parameter C1 can be selected ordinary (C1=0) or extended (C1=1) function. The following inputs should be in unit % (0-100): REFV, I, DEADB, MAXDEV, BALREF, DBPOS, DBREF.



Function diagram for extended function (C1=1)



Function diagram for ordinary function (C1=0)

Extended Function (C1=1)

The feedback can be obtained from the actuator or the element can simulate the actuator position internally. The effective pulse duration is proportional to the control error and is calculated as follows:

$PL_{eff}=\mid DEVxKxACTT \mid / 100$, where:

DEV=Control deviation=I–REFV (%)

K=Pulse length factor

ACTT=Actuator delay time interval in seconds

To permit gradual correction of a large control error, $\mid DEV \mid$, is limited to MAXDEV. The actual time pulse duration is calculated as:

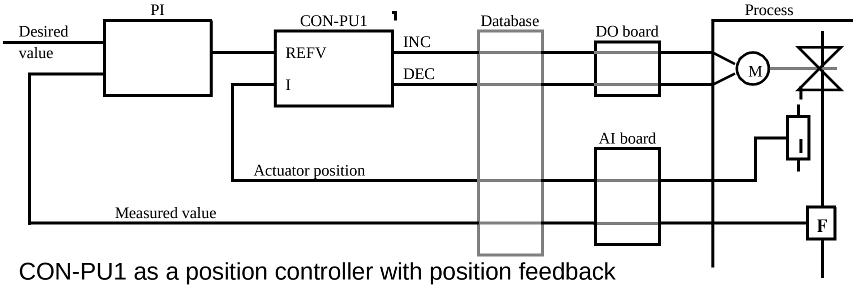
$PL=(\mid DEVxKxACTT \mid) / 100+TGAP+TPMIN$ where:

TPMIN=Min pulse duration for excitation of the shifting motor or solenoid valve. This is the shortest pulse length which affects the actuator.

TGAP=Play compensation. Added to the first pulse after a polarity change.

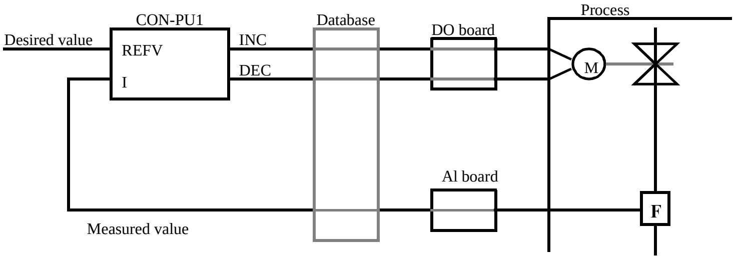
For control errors less than a deadband (DEADB, %) the correction is considered to be complete and no further pulse is calculated. TMBC is a delay time that gives the shortest pause time after a change of the pulse outputs INC/DEC.

Calculation of a new pulse duration is activated at the interval PF x TS where TS is the sample time of the function, i.e. the cycle time of the element.

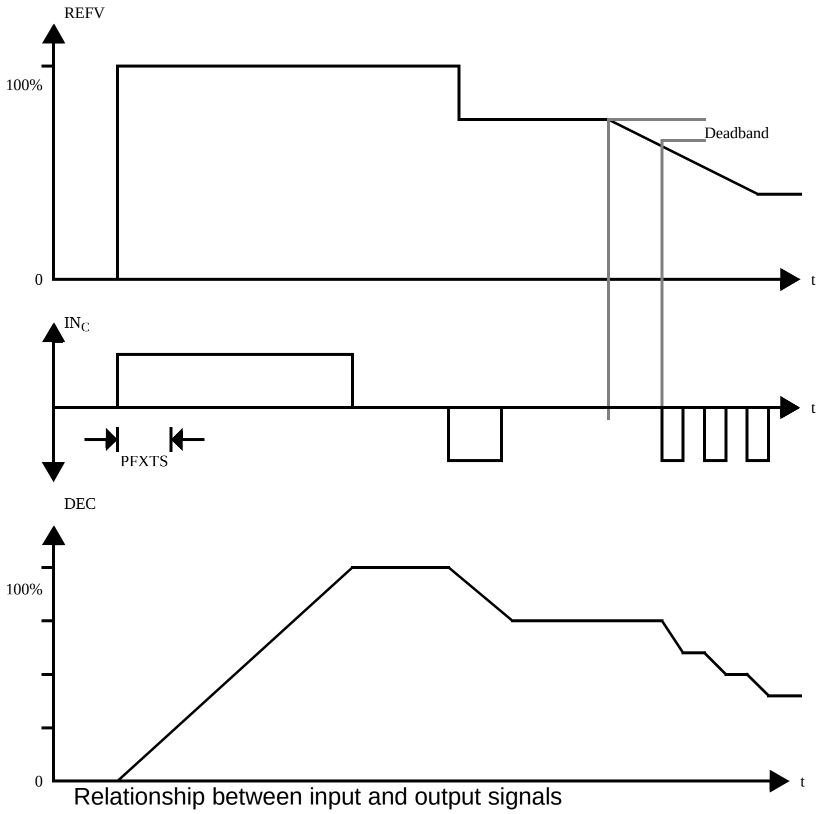


CON-PU1 as a position controller with position feedback

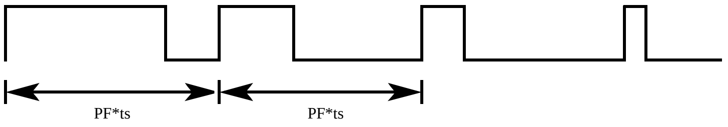
The least permitted pause time between two pulses is determined however by the parameters TPAMIN and TPAC so that the Pause time=TPAMIN+TPAC x PL where PL is the pulse duration calculated most recently. These possibilities of setting the pause time are normally only used when CON-PU1 is used as a stand alone regulator and provide great flexibility in trimming the regulator. By setting PF x TS relatively large and TPAMIN=TPAC=0, a constant pulse frequency is obtained. To obtain a constant pause time between the pulses, TPAC is set to 0 and TPAMIN to the required pause time. This is necessary when controlling processes with long time constants.



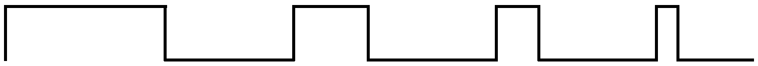
CON-PU1 as a stand alone controller



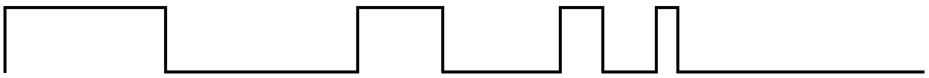
Relationship between input and output signals



Pause time = TPAMIN + TPAC x PL, TPAMN =TPAC = 0



Pause time = TPAMIN + TPAC x PL, TPAC = 0



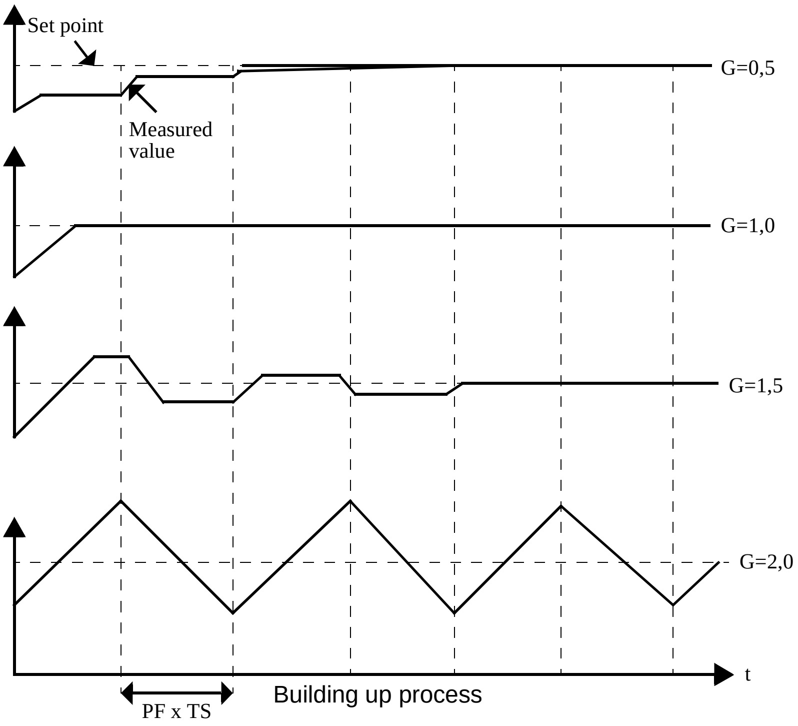
Pause time=TPAMIN + TPAC × PL. TPAMIN and TPAC > > 0

In processes with long delay times, pause times ≥ the delay time are selected.

The pause time can be made to be dependent on the duration of the latest pulse calculated with the parameter TPAC. This function is used when a rapid control of small errors is required but when limits on the speed of change in the process response require long pause times with large changes.

When CON-PU1 is used as a position regulator for a motor actuator, different building up processes are obtained depending on a parameter G, this being in principle the relationship between the real data of the actuator and the parameters set in CON-PU1.

$G=(k \times ACTT)/ACTT_v+100 \times (TGAP-TGAP_v+TPMIN-TPMIN_v)/\ (ACTT_v \times |\ DEV\ |)$, where $TGAP_v$ is the actuator play., $TPMIN_v$ the least pulse length to which the actuator reacts and $ACTT_v$ the time delay time interval of the actuator. The figure below shows the building up process for different G, where only $(k \times ACTT)/ACTT_v$ has been included. The system is unstable when G≥2.



Simulation of Actuator Position

When it is not possible to feed back the actuator position to the element, the position can be simulated internally in the element by integrating DEV × K × ACTT. The simulated value is limited to between 0 and 100%. When BAL is set, BALREF is used as an integrator value. This is used, for example, with a limit switch on the actuator to set the integrator to a known value or forced control when manually controlling the process. With simulation, the preceding regulator should have an output signal with greater range than 0–100% (t ex–100–200%).

Supervision

When $\sum_{n=1}^{N} |DEV_n \times K_n| > DBPOS$, a test is made to determine if $|I_N - I_0|$ >DBREF. DEV_n is the deviation at the beginning of pulse n, that is the deviation which is used for calculating the pulse length. I_0 is the feedback signal before the first pulse, I_N is the feedback signal after the Nth pulse.

This means that the supervision sums DEV x K for each pulse, and when the sum has exceeded the value DBPOS a check is made (after concluded pulse time) that the feedback signal has changed by at least the value DBREF. If the feedback signal has not changed enough, ERR is set. After the check, the sum is cleared.

The sum is cleared, and ERR is reset, when DEV changes sign or when DBPOS is not exceeded after 9 pulses. Theoretically you can set DBREF=DBPOS. This means that the actuator follows the controller without delay. In practice you will have to set DBREF<DBPOS.

Ordinary Function (C1=0)

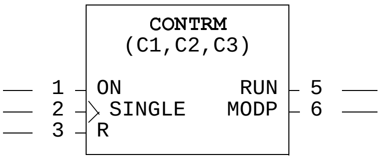
With ordinary function only pulse generation according to PL= $(|DEV| \times KI \times ACTT) / 100 + TPMIN + TGAP$ is performed.

If $|DEV| <$ DEADB no pulse is obtained.

Control Module Header CONTRM

Summary

CONTRM (**CONTR**ol **M**odule Header) is used for structuring and execution control of control modules. Function modules, blocks and elements can be included in a control module.



PC element CONTRM

Call CONTRM (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Cyclicity in ms	Allowed values are between 2ms and 20000ms in steps of 2ms
C2	Place in the cycle time table and/or scheduling strategy	1..251: place in cycle time table, cyclic activation 252: activation at initialization after power down. 254: activation at warm start initialization after power down.
C3	Currently not used	

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) with normal execution.
2	SINGLE	IB	If input ON and R are 0 the elements in the control module can be executed once by setting **SINGLE**.
3	R	IB	**R**eset. Input for clearing of the elements in the control module.
5	RUN	OB	**RUN** indicates that the elements in the control module are executed.
6	MODP	OB	**MOD**ify **P**ermission. Output that indicates if the module may be modified from service aids.

Function

The control module header CONTRM is a superior execution controlling element for a control module. The call parameter C1 is used to specify how often the control module is to be executed and call parameter C2 is used to specify the place in the cycle time table on which the execution sequence of execution units with the same cycle time is based. When and how the control module is to be executed is determined with the control inputs ON, SINGLE and R.

Normal Execution

The input ON is to be set with normal execution and the control module is then executed in accordance with the call parameters C1 and C2. If the ON input is reset after execution of the control module, the calculated data remains until the next time the control module is executed. The ON input overrides the SINGLE input, i.e. if ON is set SINGLE has no effect. The execution is performed in three steps:

1. Entry of variables from the data base and from modules with higher priority.
2. Element execution.
3. Reading of variables to the database and to modules with higher priority.

SINGLE Execution

If the SINGLE input is set when the ON input is reset, the control module is executed once in accordance with normal execution.

Clearing

If input R is set, the control module is executed in the reset mode. This means that all outputs of the elements within the control module are given original values which in most cases are the 0-value of the data. Reading to the data base is performed with data value=0 for all data types. Input R overrides the inputs ON and SINGLE.

RUN

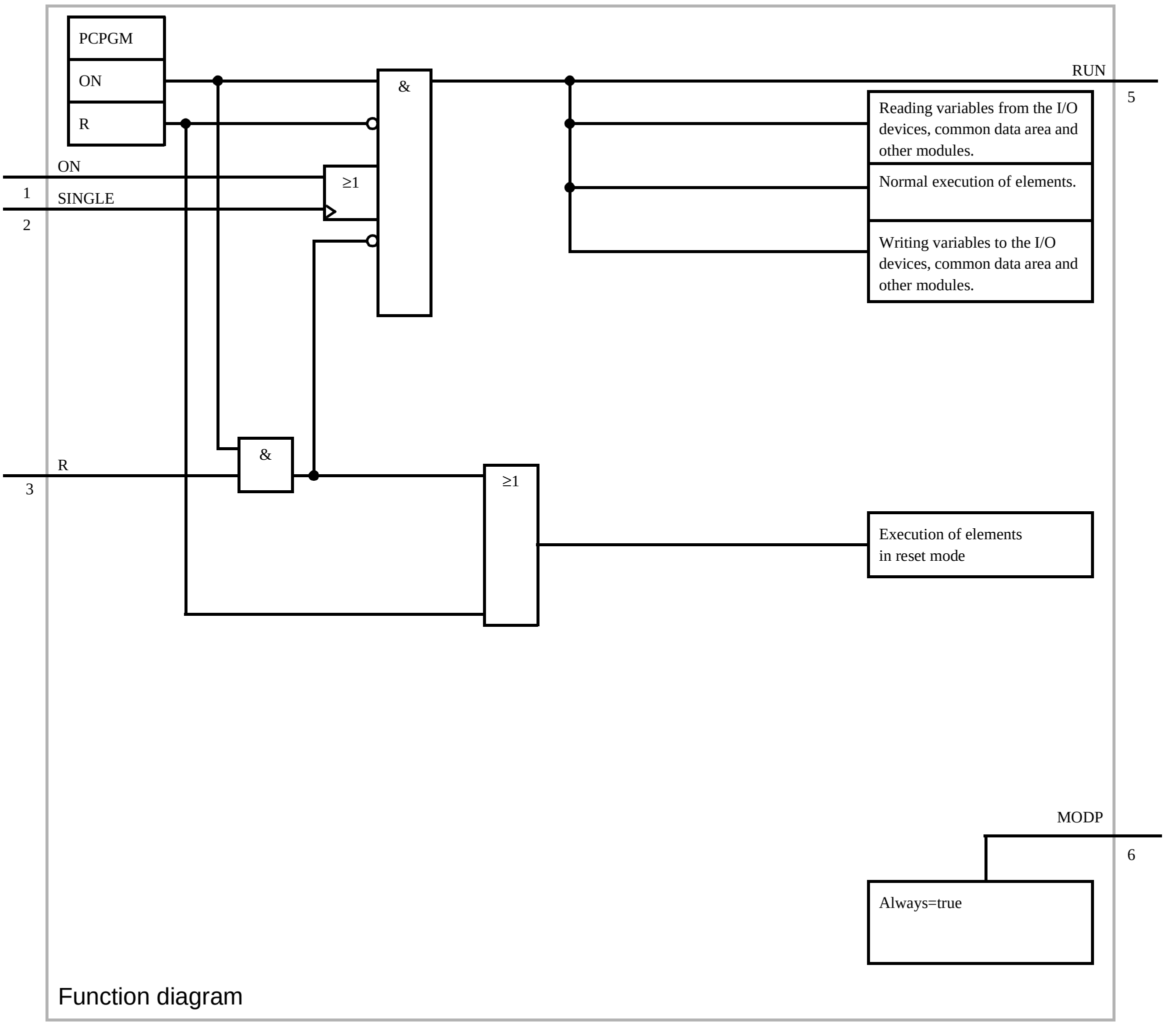
Output RUN is set only if normal execution is in progress, i.e. if ON is set, or if the control module is being SINGLE executed. With SINGLE execution, the RUN output is set during one cycle only.

MODP

Is always true.

Effects from PCPGM

ON and the R inputs and the blocking functions on the program header override the inputs on the control module header. The input ON on PCPGM must be set to permit normal execution or reset execution of the elements in the control module.

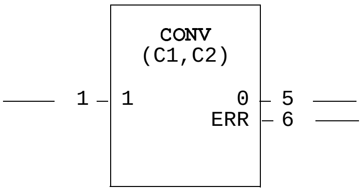

PCPGM
ON
R
&
RUN
5
Reading variables from the I/O devices, common data area and other modules.
ON
1
≥1
Normal execution of elements.
SINGLE
2
Writing variables to the I/O devices, common data area and other modules.
&
R
3
≥1
Execution of elements in reset mode
MODP
6
Always=true

Function diagram

Code Converter CONV

Summary

CONV (**CONV**erter) converts integer, real, time and time-real data from one representation to each other.



PC element CONV

Call CONV (C1, C2)

Parameter	Significance	Permissible values
C1	Data type of input	I, IL, R, T, TR
C2	Data type of output	I, IL, R, T, TR

Terminal Description

No	Name	Type	Description
1	I	I*C1*	**I**nput. Input for data to be converted.
5	O	O*C2*	**O**utput. Output for converted data.
6	ERR	OB	**ERR**or. Output which is set (to 1) if a limit is reached at output O.

Function

Data at input I with data type in accordance with call parameter C1 is converted to data with data type according to call parameter C2 at the output O. When conversion is to the data types I and IL (integer) and T (time), overflow can occur at the max or the min limit values. The error signal output ERR is set when overflow occurs.

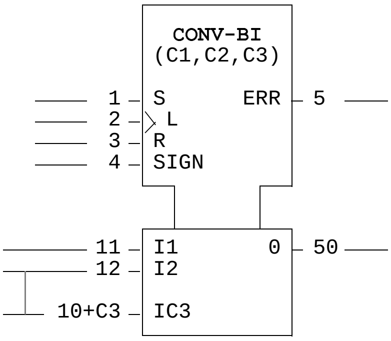
Rounding Off

When converting from real numbers to integers, the number is rounded up when the decimal is 0.5 or greater, and down when the decimal is less than 0.5. When converting from T to R, I or IL, the value given is the number of entire seconds from midnight. When converting from R, I, or IL to T, the input value is treated as the number of entire seconds from midnight.

Code Converter, Boolean to Integer CONV-BI

Summary

CONV-BI (**CONV**erter **- B**oolean to **I**nteger) converts data from BC, BCD, 1-of-N, Gray code or 16/32 Bit Packing, formed from Boolean variables into integers. The element has a memory function for storage of converted data.



PC element CONV-BI

Call CONV-BI (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type of output	I, IL
C2	Type of conversion 1 for BC 2 for BCD 3 for 1-of-N code 4 for Gray code 5 for Bit Packing(1)	1..5
C3	Number of inputs for data to be converted. If C2=5 then C3 must be 15 or 31.	1..31

(1) When C2=5 the output O cannot be interpreted as I/IL by the aid.

Number of inputs required with

Code	Number of inputs required with	
	I	IL
BC	15	31
BCD	18	31
1-of-N	31	31
Gray	15	31
Bit packing	15+ SIGN	31+SIGN

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for storage of a new value each time the element is executed. When this input is reset (to 0) the latest calculated value will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of data.
3	R	IB	**R**eset. Input for clearing of the output. This input overrides the S and L inputs.
4	SIGN	IB	Input set (to 1) with negative data values. If C2=5 SIGN is used as the most significant input to get a 16/32 bit conversion.
5	ERR	OB	**ERR**or. Output which is set when more than one input is set with conversion of 1-of-N code or if the integer value to be stored at the output cannot be represented by the data type.
11	I1	IB(1)	**I**nput **1**.
12	I2	IB(1)	**I**nput **2**.
: 10+C3	I*C3*	IB(1)	**I**nput ***C3***.
50	O	O*C1*	**O**utput. Output for converted data.

(1) The data at each of these inputs is of Boolean type but together they form a value of the type specified by parameter C1.

Function

The element converts binary code (BC), binary coded decimal code (BCD), 1-of-N code, Gray code, or uncoded Booleans (Bit Packing) to integers of data type I (16 bit) or IL (32 bit). The SIGN input can be used as a sign bit (C2=1, 2 or 3) or as most significant bit (C2=5).

Examples of Input Values with Different Decodings

Input values with different decodings

Inputs	Input integer value			
	BC with sign	BC without sign	BCD	1-of-N
I1	1	1	1	1
I2	2	2	2	2
I3	4	4	4	3
I4	8	8	8	4
I5	16	16	10	5
I6	32	32	20	6
I7	64	64	40	7
I8	128	128	80	8
I9	256	256	100	9

Input values with different decodings

Inputs	Input integer value			
	BC with sign	BC without sign	BCD	1-of-N
I10	512	512	200	10
: I31	: 2^{30}	: 2^{30}		
SIGN	sign	2^{31}		

Only one of the inputs may be set with 1-of-N code. All inputs have a fixed value with BC, BCD and 1-of-N codes but not with Gray code.

BC, I				Gray code, I				1-of-N-code, I														
4	**3**	**2**	**1**	**4**	**3**	**2**	**1**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	1	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	1	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Storage of Data

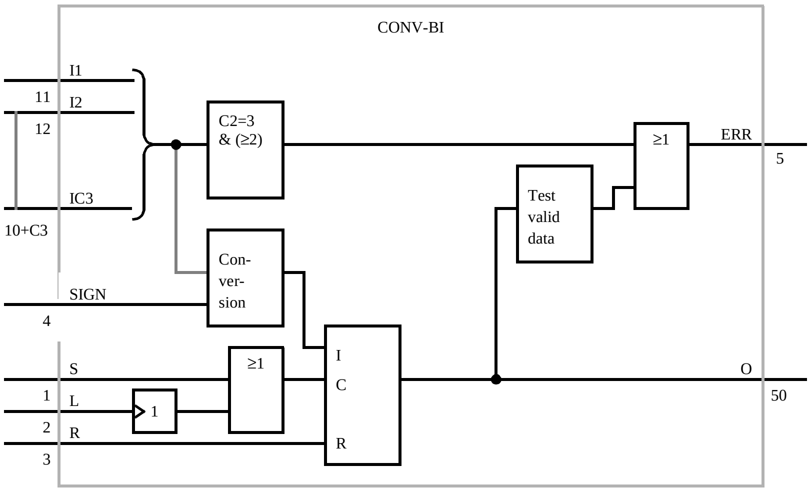
The code at the inputs I1 … I*C3* is converted and stored immediately the input L is set. If input S is set, the input code is converted and the integer is stored each time the element is executed. When S is reset (to 0) after having been set, the data stored most recently remains. The input S overrides the input L, i.e. when S is set, L has no effect.

Clearing

The input R clears the output and prevents all further storage of data while R is set.

Supervision

When converting 1-of-N code the setting of only one of the inputs 1 … I*C3* is supervised. If two or more inputs are set, the value of the input signal with the lowest number is stored. The error signal output ERR is also set. If the integer value to be stored at the output exceeds the value which can be represented, the error signal output is set. In addition, the output is limited to the upper or lower limit value.

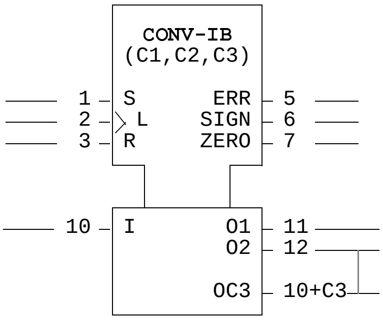


Function diagram

Code Converter, Integer to Boolean CONV-IB

Summary

CONV-BI (**CONV**erter **- I**nteger to **B**oolean) converts data from integer into BC, BCD, 1-of-N, Gray code 16/32 bit BC formed from Boolean variables. The element has a memory function for storage of converted data.



PC element CONV-IB

Call CONV-IB (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type	I, IL
C2	Type of conversion 1 for BC 2 for BCD 3 for 1-of-N code 4 for bit unpacking(1)	1..4
C3	Number of converted data.	1..32

(1) When C2=4 the input 1 cannot be interpreted as I/IL by the tool.

Number of outputs required with different codes

Code	Number of outputs required with	
	I	IL
BC	15	31
BCD	18	32
1-of-N	31	32
Bit packing	15+ SIGN	31+SIGN

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for storage of a new value each time the element is executed. When this input is reset (to 0) the latest calculated value will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of data.

Terminal Description

No	Name	Type	Description
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	ERR	OB	**ERR**or. Output which is set when the integer value to be converted exceeds the value which can be represented at the outputs.
6	SIGN	OB	Output set to 1 with negative data values. If C2=4 SIGN is interpreted as bit 16/32.
7	ZERO	OB	Output set to 1 when the value at the input I is zero.
10	I	IC1	**I**nput. Input for the integer value to be converted.
11	O1	OB(1)	**O**utput **1**.
12	O2	OB(1)	**O**utput **2**.
: 10+C3	O*C3*	OB(1)	**O**utput ***C3***.

(1) The data at each of these outputs is of Boolean type but together they form a value of the type specified by parameter C2.

Function

The element converts integer values of the data types I or IL into Boolean variables using binary code (BC), binary coded decimal code (BCD), 1-of-N code or uncoded unpacking. The code conversion to be performed is specified by the call parameters C1 and C2. With positive data values the output SIGN is reset (to 0) and with negative values, set (to 1). For bit unpacking SIGN is used as the most significant bit.

Examples of Output Values with Different Decodings

Output	Output integer value		
	BC	BCD	1-of-N-code
O1	1	1	1
O2	2	2	2
O3	4	3	3
O4	8	4	4
O5	16	5	5
O6	32	6	6
O7	64	7	7
O8	128	8	8
O9	256	9	9
O10	512	10	10
: SIGN(1)	: 2^{31}		

(1) SIGN is used as a sign for BC, BCD and 1-of-N.Only one of the outputs is set with 1-of-N code.

Storage of Data

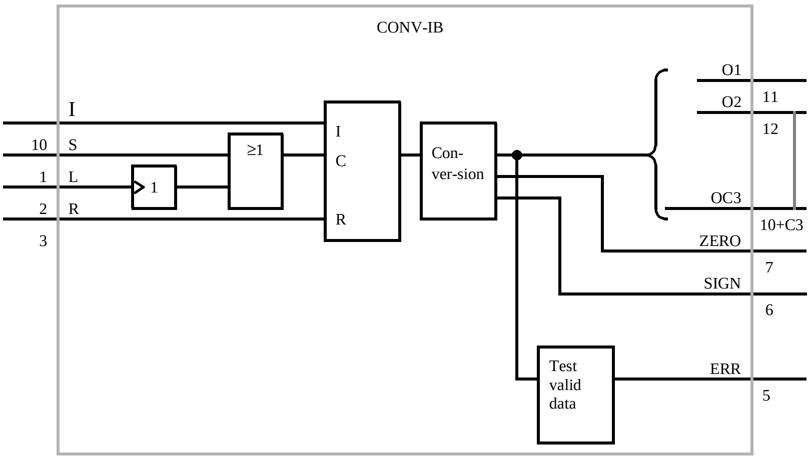
The integer at the input I is converted immediately the input L is set. If input S is set, the input code is stored and the result is converted each time the element is executed. When S is reset (to 0) after having been set, the data stored most recently remains until the element is once more executed with one of the inputs S, L or R set. The input S overrides the input L, i.e. when S is set, L has no effect.

Clearing

The input R clears the output and prevents all further storage of data while R is set.

Supervision

When converting data from integers to 1-of-N code, the numbers are restricted to –32 to +32. If the integer value to be converted exceeds the value which can be represented at the outputs, the error signal output is set. In addition, the output is limited to the upper or lower limit value.

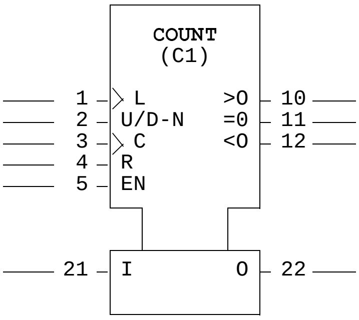


Function diagram

Counter **COUNT**

Summary

COUNT (**COUNT**er) is a presettable counter for counting pulses, up or down. The counter also monitors the relation of the counter value to 0.



PC element COUNT

Call COUNT (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description
1	L	IB	**L**oad. **L**oads the counter with the value at input I.
2	U/D- N	IB	**U**p/**D**own-**N**. Input which determines if the counter is to count up (U/D-N=1) or down (U/D-N=0).
3	C	IB	**C**lock. When this input changes from 0 to 1 the counter counts up or down in accordance with the status of the input U/D-N.
4	R	IB	**R**eset. Input which clears the counter and prevents all further counting or loading.
5	EN	IB	**EN**able. Input which is to be 1 to permit counting or loading. Reset execution is, however, performed independently of EN.
10	>0	OB	Output which is set when the value of the counter is greater than 0.
11	=0	OB	Output which is set when the value of the counter is 0.
12	<0	OB	Output which is set when the value of the counter is less than 0.
21	I	I*C1*	**I**nput. Input for new value when loading.
22	O	O*C1*	**O**utput. Output for counter value.

Function

The counter value increases or decreases by 1 immediately the input C is set. The value increases if U/D-N=1 and decreases if U/D-N=0. The duration of the counter period may not be less than 2 x the cycle time of the program.

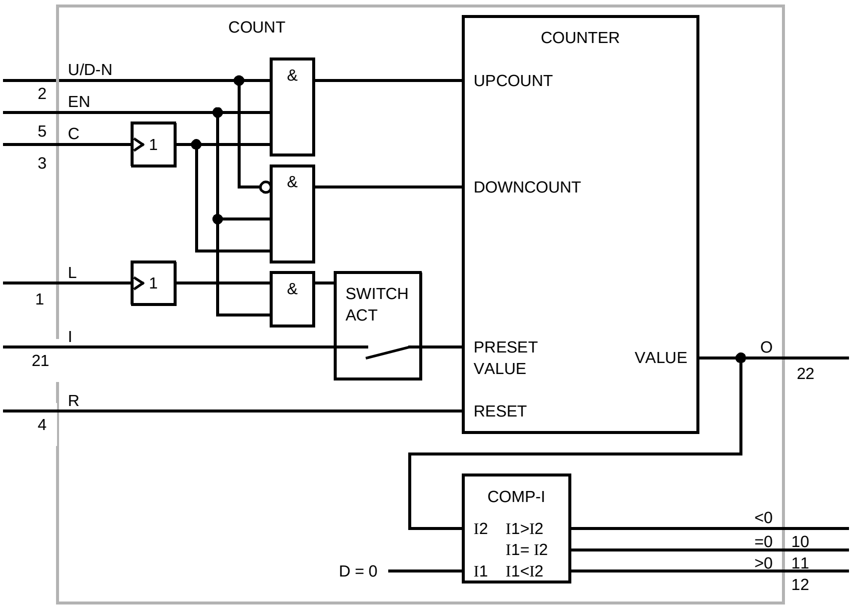
When the input L is set, the counter is loaded with the value at input I. If both input L and input C are set simultaneously, the counter is first loaded after which an up or down count is performed. The input EN must be set for the counter to count or load a new value

Clearing

The input R clears the counter and prevents all further counting or loading. R overrides EN.

Supervision

The status outputs specify the relation of the counter value to zero (>0, =0, <0). When the counter reaches its least or greatest value for the data type, all counting ceases.

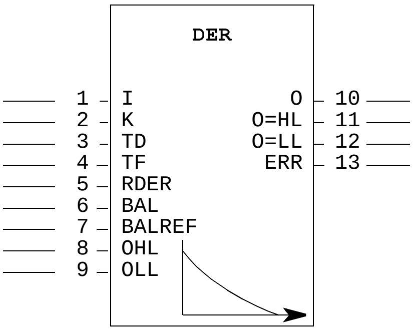


Function diagram

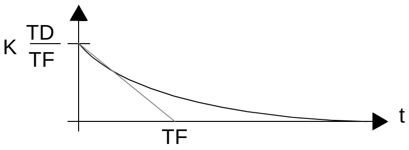
Derivator DER

Summary

DER (**DER**ivator) is used to give derivation effect. The derivation effect can be limited with the filter function, which serves as a low pass filter. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.



PC element DER



Step response

Call DER

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	TD	ITR	**T**ime **D**erivation. Input for time constant for derivation.
4	TF	ITR	**T**ime **F**ilter. Input for filter time constant.
5	RDER	IB	**R**eset **DER**ivator. Input for clearing derivator.
6	BAL	IB	**BAL**ance. Input for activation of following.
7	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
8	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
9	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O= HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O= LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

Function

The step response in the time plane for a derivator is:

$O(t)=K\ (TD/TF)e^{-t}/TF \times I(t)$ where I(t) specifies the magnitude of the step.

The transfer function for a DER function is:

$G(s)=K(s \times TD)/(1+s \times TF)$

This has been implemented in the DER element as a recursive algorithm. The design of the algorithm is such that normal functioning is maintained even during limiting. This ensures a controlled return to a dynamic state.

Gain, Derivation Time, Filter Time and Sampling Time

Certain constants are pre-calculated to make the execution time of the element as short as possible. The results are stored internally in the element. These constants are recalculated if TD, TF or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check whether TD and TF $\geq 2 \times TS$. If this is not the case TD and/or TF is set equal to $2 \times TS$.

Clearing of the Derivator

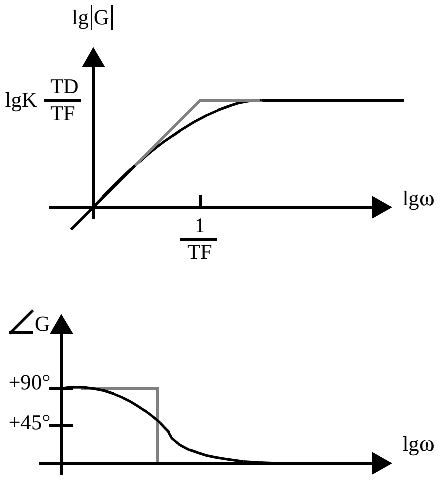
Both the output O(t) and the internal state of the output are cleared when RDER goes to 1.

Following

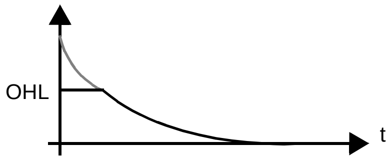
If BAL is set to 1, the derivator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

Limitation Function

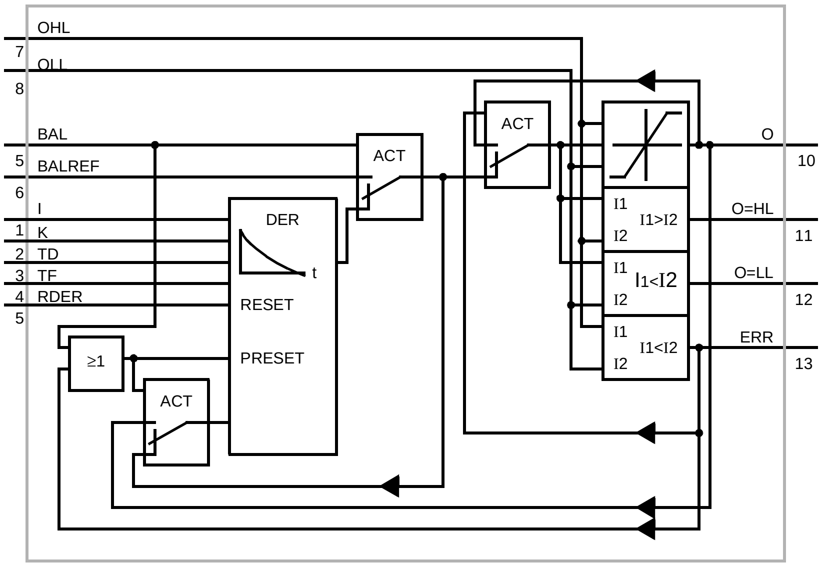
The limitation function limits the output signal to the values at the inputs OHL for upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following above.



Bode diagram



Function in a limiting state

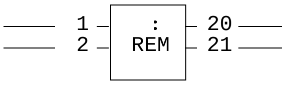


Function diagram

Divider DIV

Summary

DIV is used for division of two integers or real numbers. When dividing integers, the quotient is obtained with the remainder at a separate input.



PC element DIV

Call DIV (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	-	I*C1*	Input for dividend.
2	-	I*C1*	Input for divisor.
20	-	O*C1*	Output for quotient.
21	REM	O*C1*	**REM**ainder. Output for remainder. Applies only with division of integers.

Function

The value at input 1 is divided by the value at input 2. The quotient is stored at output 20. When dividing integers, the remainder is stored at the output REM.

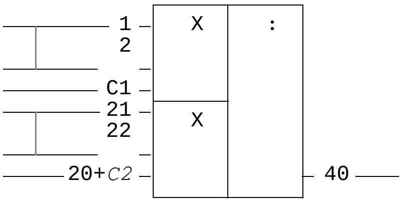
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Divider with Multiple Inputs DIV-MR

Summary

DIV-MR (**DIV**ider **-** with **M**ultipliers **R**eal numbers) is used for division of two products of real numbers.



PC element DIV-MR

Call DIV-MR (C1, C2)

Parameter	Significance	Permissible values
C1	Number of inputs in the first multiplication element.	1..19
C2	Number of inputs in the second multiplication element.	1..19

Terminal Description

No	Name	Type	Description
1	-	IR	Factor for dividend.
2	-	IR	Factor for dividend.
: *C1*	-	IR	Factor for dividend.
21	-	IR	Factor for divisor.
22	-	IR	Factor for divisor.
: 20+*C2*	-	IR	Factor for divisor.
40	-	OR	Output for quotient.

Function

The product of the real values at the inputs 1 … *C1* is divided by the product of the real values at the inputs 21 … 20+C2. The quotient is stored at the output 40.

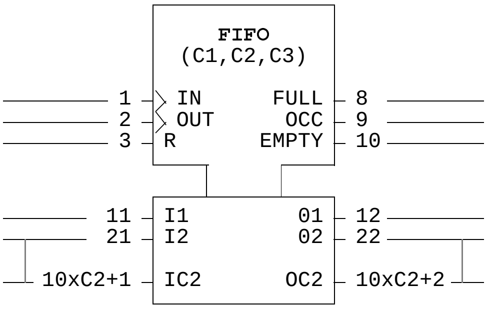
Overflow

If the maximum positive or negative real values are exceeded, the output is limited to the highest or lowest representable value respectively.

Queue Register FIFO

Summary

The queue register FIFO (**F**irst **I**n **F**irst **O**ut) can be used for storage of data to be used later and irrespective of when the data was stored. The different queues in the element work in parallel, i.e. reading in, reading out, etc. are performed at the same time in all of the queues. Data may be of the types integer, real number, Boolean or time. A maximum of 9 queues each with a maximum of 64 queue places can be selected.



PC element FIFO

Call FIFO (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of queues	1..9
C3	Number of queue places in each queue	2..64

Terminal Description

No	Name	Type	Description
1	IN	IB	Dynamic input for entry of data in the queues. Data is read from the inputs I1 … I*C2*.
2	OUT	IB	Dynamic input for reading data from the queues.
3	R	IB	**R**eset. Input which clears the queue register and prevents all further entry.
8	FULL	OB	Output which is set when the queue register is full.
9	OCC	OI	**OCC**upied. Output which specifies how many queue places are occupied.
10	EMPTY	OB	Output which is set when the queue register is empty.
11	I1	I*C1*	**I**nput **1**. Input data to queue no. 1.
12	O1	O*C1*	**O**utput **1**. Output data from queue no. 1.
21	I2	I*C1*	**I**nput **2**. Input data to queue no. 2.
22	O2	O*C1*	**O**utput **2**. Output data from queue no. 2.
: 10xC2+1	I*C2*	I*C1*	**I**nput ***C2***. Input data to queue no. C2.
10xC2+2	O*C2*	O*C1*	**O**utput ***C2***. Output data from queue no. C2.

Function

Each queue has a data input I. It is used when data is to be entered at the end of the queue. Each queue has a data output O. Data from place 1 in each queue is continuously available at the outputs O1 … O*C2.* If both of the dynamic inputs (IN and OUT) are set during the same program cycle, entry is performed first and then reading out.

Entry of Data

Data at the I inputs are placed last in the queues when input IN is set. When the queue register is empty, data is stored at queue place 1, then at queue place 2, etc.

Output of Data

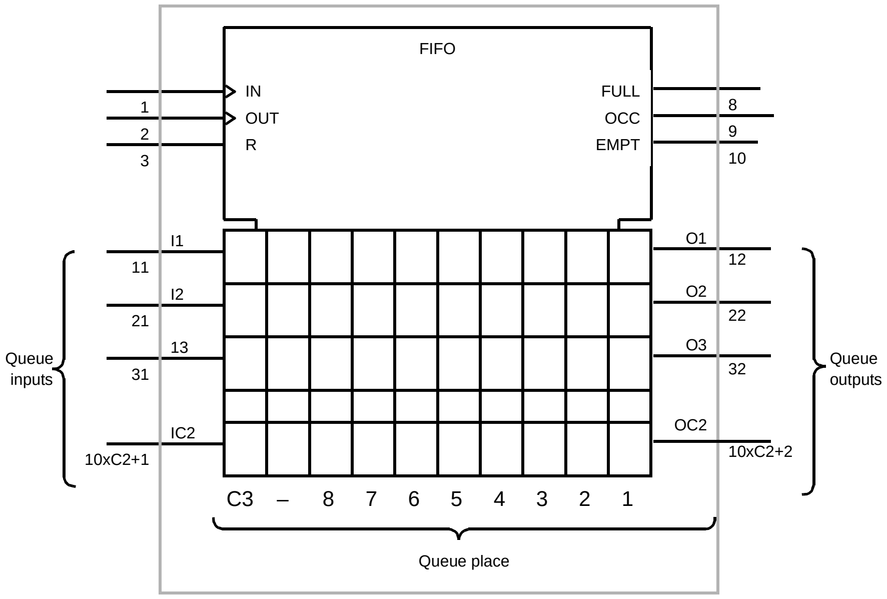
When input OUT goes to 1, data at each used place in all the queues is moved to the place with the number immediately below. 0 is written at the last place used. Data which was previously stored at queue place 2 is now at queue place 1 and can be read at the outputs O1 … O*C2*.

Clearing of the Complete Register

When the input R is set, the complete contents of the queues are cleared. All further entry is blocked.

Supervision

How many queue places are occupied is indicated continuously by the output OCC. When the queue register is full, the output FULL is set and when the queue register is empty, the output EMPTY is set.

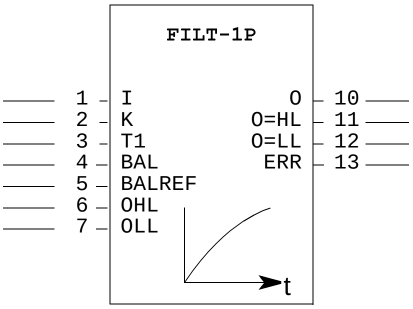


Function diagram

Filter FILT-1P

Summary

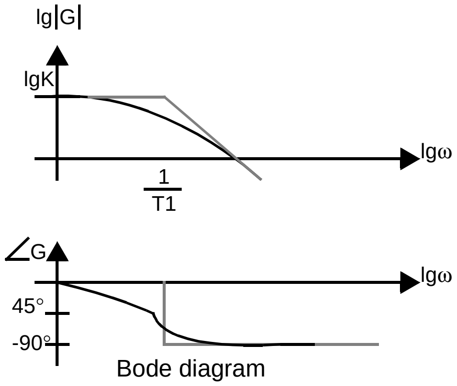
FILT-1P (**FILT**er **- 1 P**ole) is used as a single pole low pass filter. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.



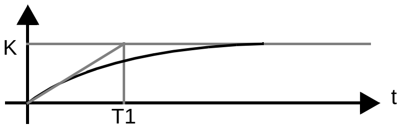
PC element FILT-1P

Call FILT-1P

Diagrams



Bode diagram



Step response

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T1	ITR	**T**ime **1**. Input for filter time constant.
4	BAL	IB	**BAL**ance. Input for activation of following.
5	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
6	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
7	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O= LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.

Terminal Description

No	Name	Type	Description
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.

Function

The step response in the time plane for a single pole low pass filter is: O(t)=I(t) K (1–e$^{-t/T1}$). The transfer function for a single pole low pass filter is: G(s)=K(1/(1+sT1)). This has been implemented in the FILT-1P element as a recursive algorithm.

Gain, Filter Time and Sampling Time

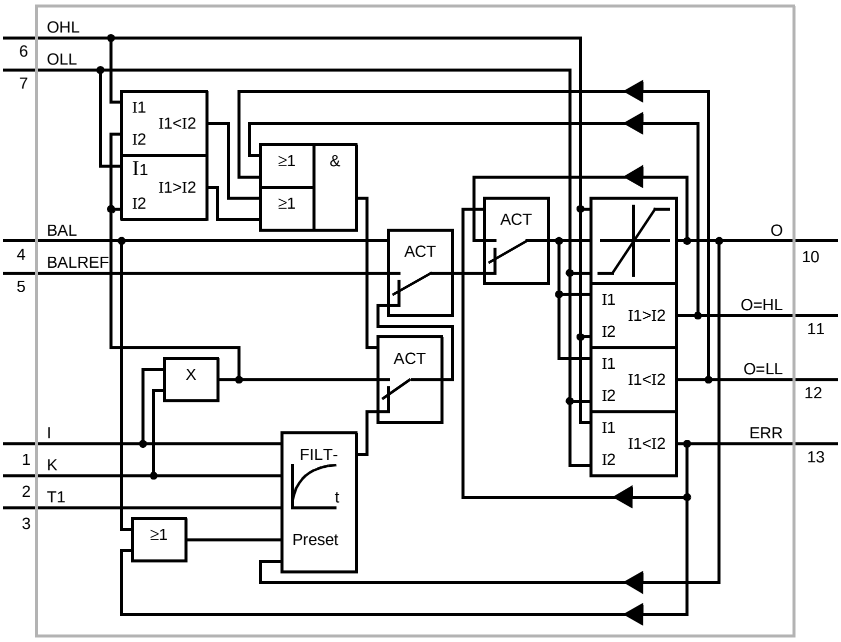
Certain constants are pre-calculated to make the execution time of the element as short as possible. The result is stored internally in the element. These constants are recalculated if T1 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating, a test is performed to check if T1>2 × TS. If this is not the case T1 is set equal to 2 × TS.

Following

If BAL is set to 1, the filter immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit value, the output O=HL is set to 1 and if it falls below the lower limit value, the output O=LL is set to 1. When the limitation status has been detected, a check is made each time the element is executed to determine whether K × I(t) exceeds the output signal limitations. If such is the case, the limitation status remains. If not, the calculation of the output signal is performed by the algorithm in the normal way. Return from limitation to a dynamic state is bumpless. The element checks that the upper limit value OHL is greater that the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following above.



Function diagram

Function Header FUNCM

Summary

FUNCM (**FUNC**tion **M**odule) is used for structuring a PC program.

PC element FUNCM

Call FUNCM

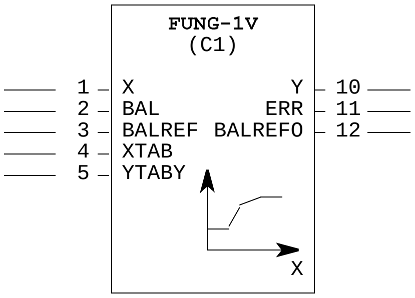
Function

The purpose of FUNCM is to control the arrangement of the documentation. The element cannot affect the execution. A function module can be used to provide a subdivision of a PC program, a control module, a slave or a sequence step in several functionally associated parts. A function module can itself contain a control module, slave sequence or a new function module.

Function Generator FUNG-1V

Summary

FUNG-1V (**FUN**ction **G**enerator - **1 V**ariable) is used for generation of an optional function of one variable, y=f(x). The function is described by a number of coordinates. Linear interpolation is used for values between these coordinates. A maximum of 255 coordinates can be specified.



PC element FUNG-1V

Call FUNG-1V (C1)

Parameter	Significance	Permissible values
C1	Number of coordinates which describe the function.	2..255

Terminal Description

No	Name	Type	Description
1	X	IR	**X**-value. Input for X-value.
2	BAL	IB	**BAL**ance. Input for activation of balancing.
3	BALREF	IR	**BAL**ance **REF**erence. Value which the Y-output is to adopt with balancing.
4	XTAB	IGC1R	**X TAB**le. Group data for the X-table with C1 values.
5	YTAB	IGC1R	**Y TAB**le. Group data for the Y-table with C1 values.
10	Y	OR	**Y**-value. Output for Y-value.
11	ERR	OB	**ERR**or. Error signal which is set (to 1) if X is outside the value of XTAB or if Y, on balancing, is outside the YTAB values.
12	BALREFO	OR	**BAL**ance **REF**erence Output. Output for calculated X-value with balancing.

Function

The function generator FUNG-1V for one variable calculates an output signal Y for a value at the input X. The calculation is performed in accordance with a piece by piece linear function which is determined by the vectors XTAB and YTAB. For each X-value in XTAB, there is a corresponding Y-value in YTAB. The Y-value at the output is calculated by means of linear interpolation between the two X-values in XTAB which are nearest the value at the input X. The values in X-tab must be strictly increasing from low to high serial numbers in the table.

Interpolation

The function generated can be illustrated by the following figure. The interpolation is performed as follows:

$y=y_k+(x-x_k)\ (y_{k+1}-y_k)/(x_{k+1}-x_k)$

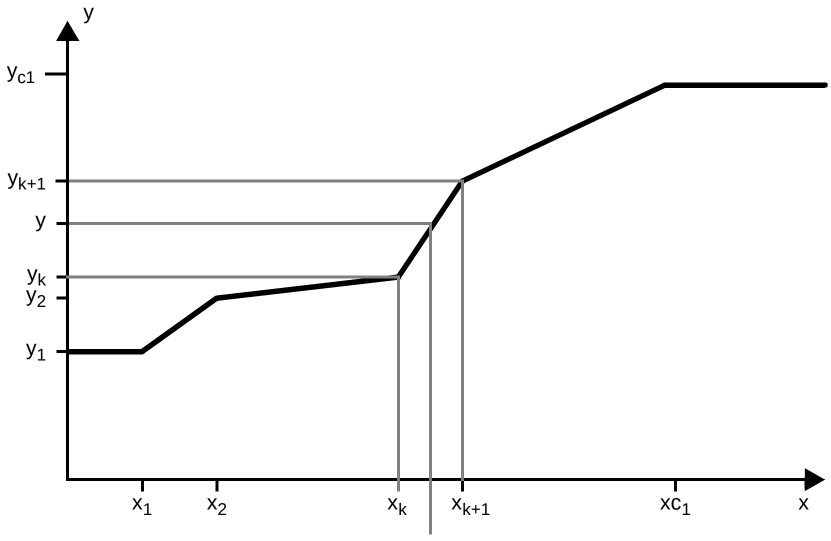
Balancing

On activation of the balancing input BAL, the value at Y is set to the value at the input BALREF. The X-value which corresponds to this Y-value is obtained at the output BALREFO. On balancing, the X-value is calculated by interpolation in the same way as the Y-value is calculated during normal operation. To permit the balancing, the values in YTAB must be strictly increasing or decreasing from low to high serial numbers in the table.

Error signal

If the input signal X is outside the range defined by XTAB, the ERR output is set to 1. The Y-value is then set to the greatest or lowest value resp. in YTAB.

ERR is also set to 1 if BALREF is outside the YTAB value range when BAL is set to 1. The value at Y is then set to the value at the input BALREF and BALREFO is set to the greatest or lowest value resp. in XTAB.



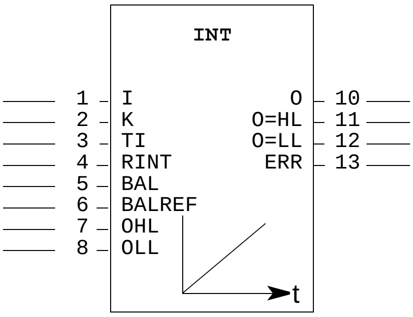
Example of function

Integrator INT

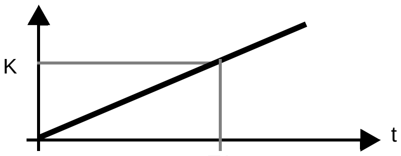
Summary

INT (**INT**egrator) is used to give an integration effect. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function.

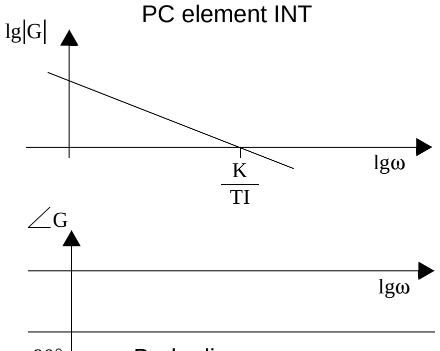
Call INT



PC element INT



Step response



Bode diagram

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input signal.
2	K	IR	Input for setting of gain.
3	TI	ITR	**T**ime **I**ntegration. Input for time constant for integration.
4	RINT	IB	**R**eset **INT**egrator. Input for clearing of integrator.
5	BAL	IB	**BAL**ance REFerence. Input for reference value when following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Input which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

Function, Transfer Function

The INT function can be written in the time plane as: $O(t)=K/TI(\int I(\tau)\, d\tau)$. The main property when controlling is that the output signal retains its value when the input signal $I(t)=0$.The step response in the time plane is $O(t)=k \times I(t) \times t/TI$. The transfer function for an integrator is $G(s)=K(1/sTI)$

Gain, Integration Time Constant and Sampling Time

The constant K × TS/TI is pre-calculated to reduce the execution time of the element to a minimum and the result is stored internally in the element. This constant is recalculated if TI or K is changed by more than 1/128 of their previous values or if the sampling time TS is changed. When recalculating a test is made to see whether TS/TI<1. TS/TI is otherwise set equal to 1.

Clearing of Integrator

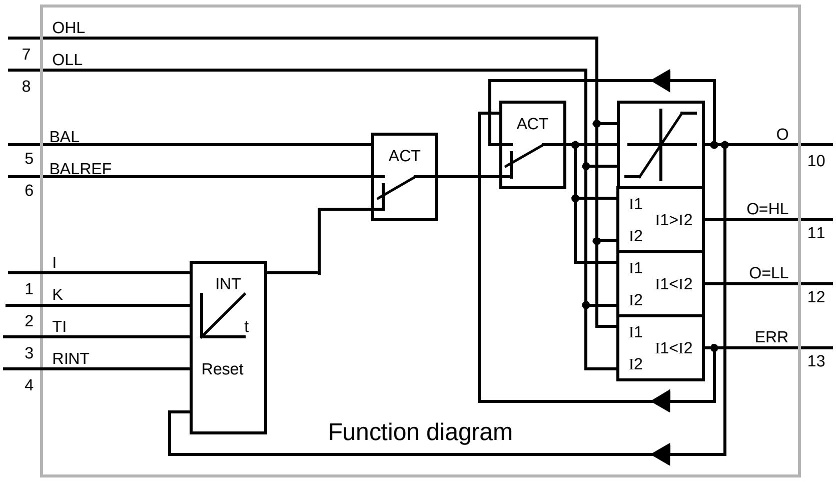
The algorithm is cleared when RINT goes to 1.

Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. On return to the normal function the value of output O during the last sample in following remains a further sample time, after which integration will be performed for this value.

Limitation Function

The limitation function limits the output signal to the values at the inputs OHL for upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After limitation or error status, normal integration is performed from the current value.



Function diagram

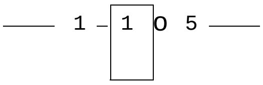
NOTE

The value of output terminal O has only seven significant digits. When integrating small increments the following problem can arise: 1000.0000+0.0001=1000.0000 and not 1000.0001.

Inverter INV

Summary

INV is used for inverting Boolean variables and particularly when reading signals to the data base.



PC element INV

Call INV

Terminal Description

No	Name	Type	Description
1	--	IB	Input.
5	--	OB	Output of inverted input value.

Function

The output signal from the INV element is set (to 1) if the input signal to the element is 0 and is reset (to 0) when the input signal is 1. See table below.

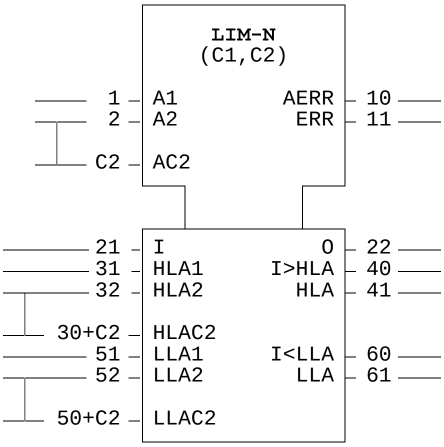
Input and Output Channels

1	5
1	0
0	1

Limiter **LIM-N**

Summary

LIM-N (**LIM**iter **-** 1-of-**N** address) is used for limitation of integers, real numbers or time values. Several limit values can be selected.



PC element LIM-N

Call LIM-N (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, T, TR
C2	Number of optional limit values	1..9

Terminal Description

No	Name	Type	Description
1	A1	IB	**A**ddress **1**. Input which, when set, limits the output O to the limit values connected to the inputs HLA1 and HLL1.
2	A2	IB	**A**ddress **2**. Input which, when set, limits the output O to the limit values connected to the inputs HLA2 and HLL2.
: C2	A*C2*	IB	**A**ddress ***C2***. Input which, when set, limits the output O to the limit values connected to the inputs HLA*C2* and LLA*C2*.
10	AERR	OB	**A**ddress **ERR**or. Output which is set when 2 or more of the inputs A1..A*C2* are set.
11	ERR	OB	**ERR**or. Output which is set when the limit for high level is less than the limit for low level.
21	I	I*C1*	**I**nput. Input to which the signal to be limited is connected.
22	O	O*C1*	**O**utput. Output for the limited signal.
31	HLA1	I*C1*	**H**igh **L**imit **A**ddress **1**. Input for upper limit value which limits the signal when the input A1 is set.

Terminal Description

No	Name	Type	Description
32	HLA2	I*C1*	**H**igh **L**imit **A**ddress **2**. Input for upper limit value which limits the signal when the input A2 is set.
: 30+C2	HLA*C2*	I*C1*	**H**igh **L**imit **A**ddress ***C2***. Input for upper limit value which limits the signal when the input AC2 is set.
40	I>HLA	OB	**I**nput>**H**igh **L**imit **A**ddress. Output which is set when the upper limit of the element is reached.
41	HLA	O*C1*	**H**igh **L**imit **A**ddress. Output which specifies the upper limit where limiting begins.
51	LLA1	I*C1*	**L**ow **L**imit **A**ddress **1**. Input for lower limit value which limits the signal when the input A1 is set.
52	LLA2	I*C1*	**L**ow **L**imit **A**ddress **2**. Input for lower limit value which limits the signal when the input A2 is set.
: 50+C2	LLA*C2*	I*C1*	**L**ow **L**imit **A**ddress ***C2***. Input for lower limit value which limits the signal when the input A*C2* is set.
60	I<LLA	OB	**I**nput<**L**ow **L**imit **A**ddress. Output which is set when the lower limit of the element is reached.
61	LLA	O*C1*	**H**igh **L**imit **A**ddress. Output which specifies the lower limit where limiting begins.

Function

LIM-N is used to limit up to 9 different limits. Boolean output signals are given when the output is limited.

Selection of Limit Value

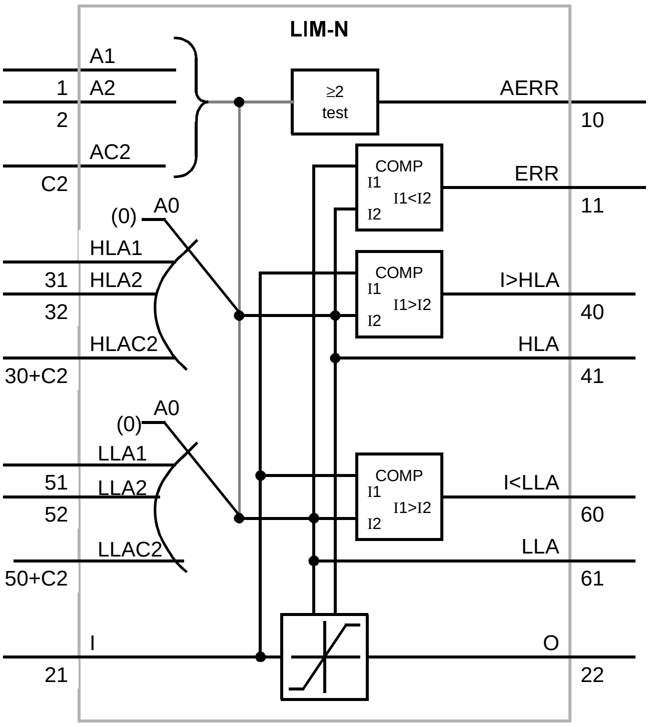
Which of the limit value inputs HLA1 … HLA*C2* or LLA1 … LLA*C2* is to limit the value at output O is selected with the inputs A1 … A*C2*. If the input A1 is 1, the output is limited by HLA1 and LLA1, if A2 is 1, the output is limited by HLA2 and LLA2, etc. If none of the inputs A1 … A*C2* is 1, O is limited to the data value 0. If 2 or more of the inputs A1..A*C2* are set at the same time, the output is limited by the limit values corresponding to the lowest numbered set input. The error signal output AERR is set at the same time.

Limiting

When the input I exceeds the selected limit, the output O is limited to the limit value. One of the outputs I>HLA or I<LLA will then be set depending on which limit was exceeded. The value of the current limits for high and low level where limiting begins can be read at the outputs HLA and LLA.

Supervision of Limit Values

The element checks that the limit value HLA is greater than the limit LLA. If HLA is less than LLA, the error signal output ERR is set. The output O and the limit value outputs I>HLA and I<HLL are retained from the sample before that in which the error status developed.

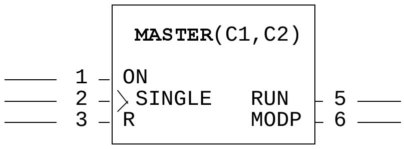


Function diagram

Master Header **MASTER**

Summary

The master header MASTER is used for the execution control of subordinate slave modules.



PC element MASTER

Call MASTER (C1, C2)

Parameter	Significance	Permissible values
C1	Cyclicity in ms	Allowed values are between 2ms and 20000ms in steps of 2ms
C2	Place in the cycle time table and/or scheduling strategy	1..251: place in cycle time table, cyclic activation 252: activation at initialization after power down. 254: activation at warm start initialization after power down.

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) with normal execution.
2	SINGLE	IB	If input ON and R are 0 the elements in the slave modules can be executed once by setting SINGLE to 1.
3	R	IB	**R**eset. Input for clearing of the slave modules.
6	MODP	OB	**MOD**ify **P**ermission. Output that indicates if the master and slave modules may be modified from service aids.

Function

MASTER is a superior execution controlling element for a number of slave modules. How often the master and associated slave modules are to be executed is determined with the call parameter C1, and the place in the cycle time table is determined with the call parameter C2. The sequence in which execution units with the same cycle time are to be executed is based on the place in the cycle time table. When and how the slave modules are to be executed is determined by the control inputs ON, SINGLE and R. The master is always executed regardless of the state of these inputs.

Normal Execution

For normal execution the input ON must be set to 1. The master module and associated slave modules are then executed as determined by the call parameters C1 and C2. The order in which the subordinate slave modules are to be executed is determined by their order in the documentation.

If the ON input is reset after execution of the slave modules, the calculated data remains until the next execution of the slave modules. The ON input overrides the SINGLE input, i.e. if ON is set to 1. SINGLE has no effect. The slave modules are executed successively in three steps:

1. Entry of variables from the data base and from modules with higher priority.

2. Element execution.

3. Reading of variables to the data base and to modules with higher priority.

SINGLE Execution

If the SINGLE input is set when the ON input is reset, the slave modules are executed once as described above. The calculated data remains until the next execution of the master and slave modules.

Clearing

If the input R is set, the slave modules are executed in the reset mode. This means that all outputs of the elements in the slave modules are given initial values, which in most cases are the 0-value of the data concerned. Input R overrides the inputs ON and SINGLE

RUN

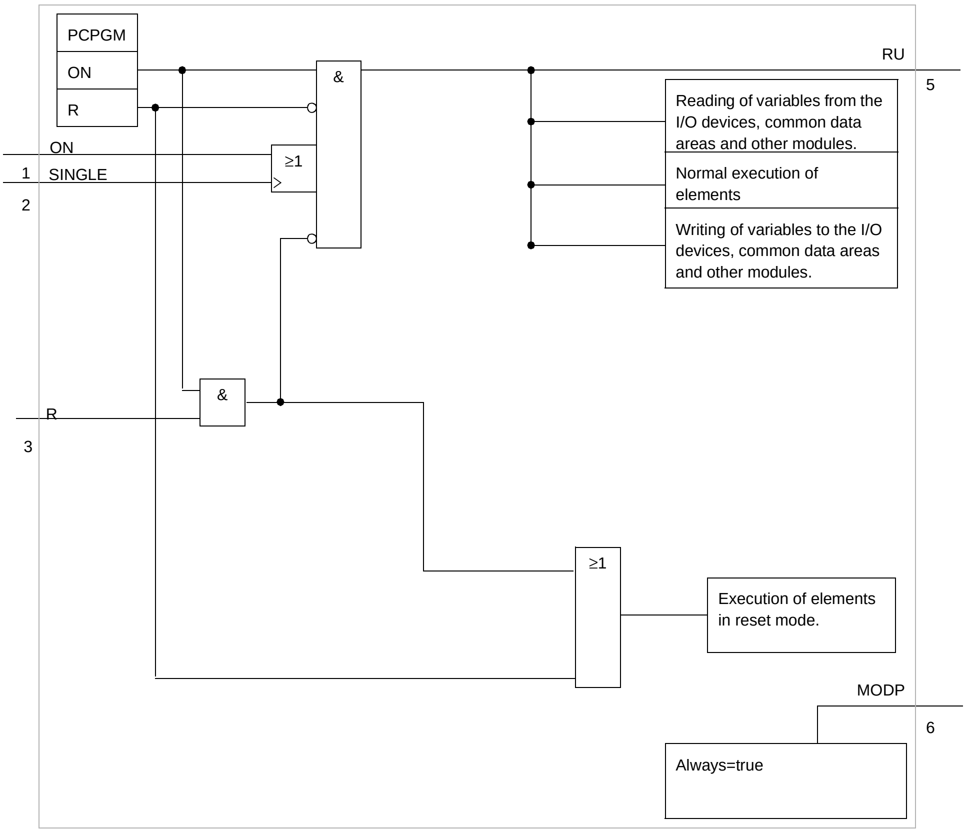
The output RUN is set only during normal execution, i.e. ON is set or SINGLE execution is performed. With single execution, the RUN output is set for only one cycle.

MODP

Always true.

Effects from PCPGM

The ON and R inputs and the blocking functions on the program header override the inputs on the master module header.
The input ON on PCPGM must be set to permit normal execution of master and slave modules.

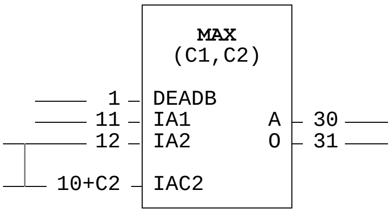

PCPGM
ON
R
RU
5
&
ON
1
SINGLE
2
≥1
Reading of variables from the I/O devices, common data areas and other modules.
Normal execution of elements
Writing of variables to the I/O devices, common data areas and other modules.
&
R
3
≥1
Execution of elements in reset mode.
MODP
6
Always=true

Function diagram

Maximum Selector MAX

Summary

MAX (**MAX**imum selector) is used to select the largest value of an optional number of integers or real numbers.



PC element MAX

Call MAX (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2..19

Terminal Description

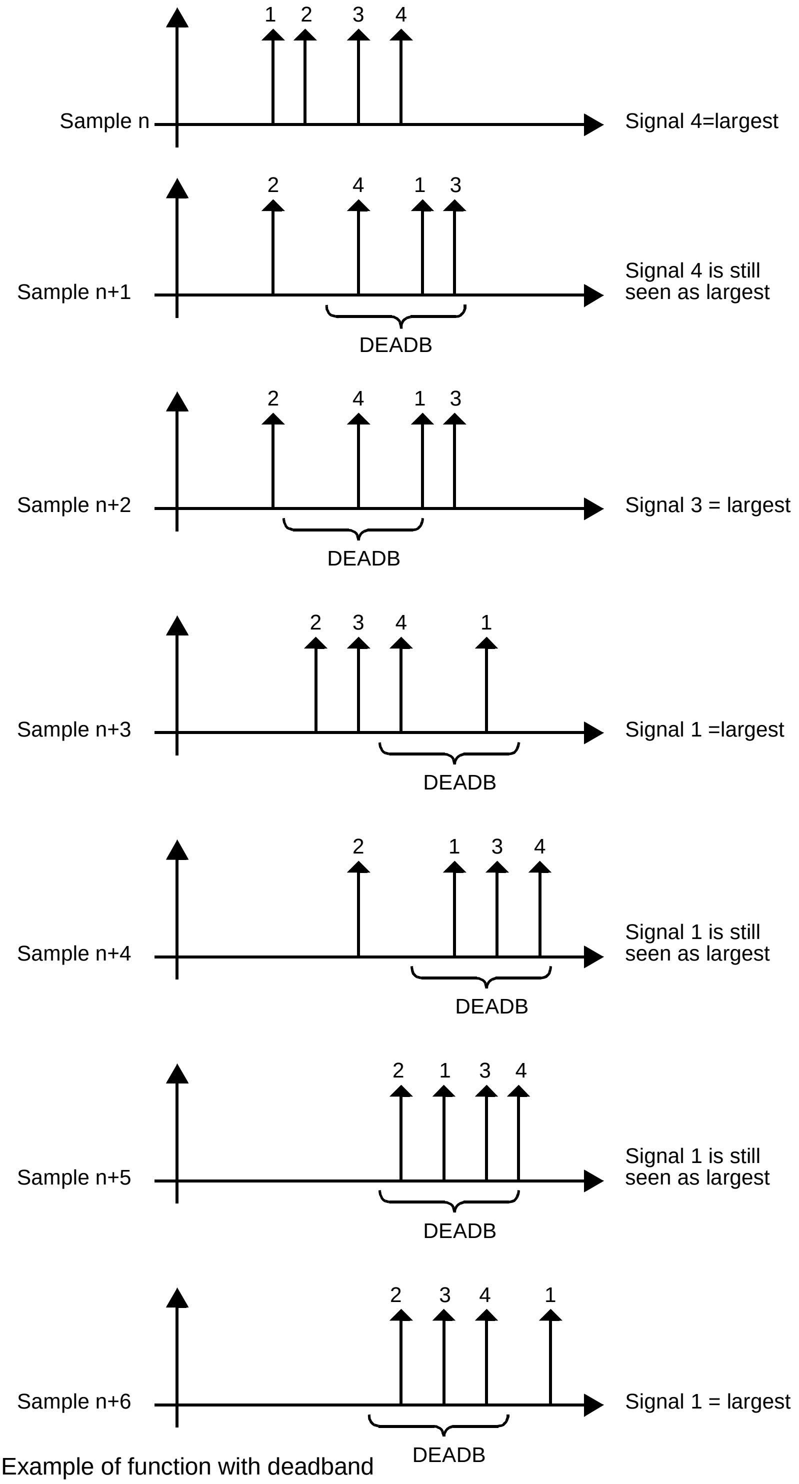
No	Name	Type	Description
1	DEADB	I*C1*	**DEADB**and. Input for deadband.
11	IA1	I*C1*	**I**nput **A**ddress **1**. Input the value of which is compared with the other inputs.
12	IA2	I*C1*	**I**nput **A**ddress **2**. Input the value of which is compared with the other inputs.
: 10+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. Input the value of which is compared with the other inputs.
30	A	OI	**A**ddress. Output for the number of the input having the greatest value.
31	O	O*C1*	**O**utput. Output for the greatest value.

Function

The values at the inputs IA1 … IA*C2* are compared and the greatest value is obtained at the output O. The number of the input with the greatest value is obtained at the output A. If the two largest signal values are equal when the element is executed the first time, the signal with the lowest connection number is selected as the greatest.

Deadband

The deadband specified at the input DEADB is symmetrical around the value of the greatest input. The upper and lower deadband limits are calculated from the value for the largest input in the preceding sample. To prevent rapid changes at the output A, the value at A is retained until the value at the corresponding input is less than the calculated lower deadband limit or until one of the other inputs exceeds the upper deadband limit.

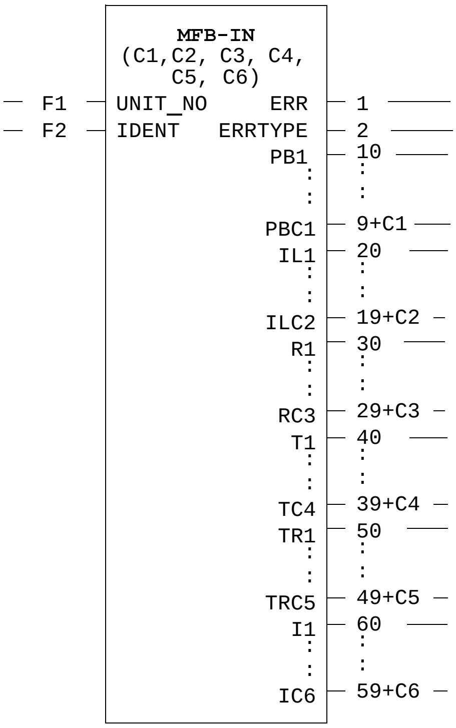


Example of function with deadband

Input from MasterFieldbus MFB-IN

Summary

The element MFB-IN is used for transferring data to Advant Controller 110 from an Advant Controller 400 Series or a MasterPiece 200/1 connected via MasterFieldbus.



PC element MFB-IN

Call MFB-IN (C1, C2, C3, C4, C5, C6)

Parameter	Significance	Permissible values
C1	Number of packed booleans	0..9
C2	Number of IL values	0..9
C3	Number of Real values	0..9
C4	Number of Time values	0..9
C5	Number of TimeReal values	0..9
C6	Number of I values	0..19
$2\times C1+2\times C2+2\times C3+2\times C4+2\times C5+C6\leq 19$		

Terminal Description

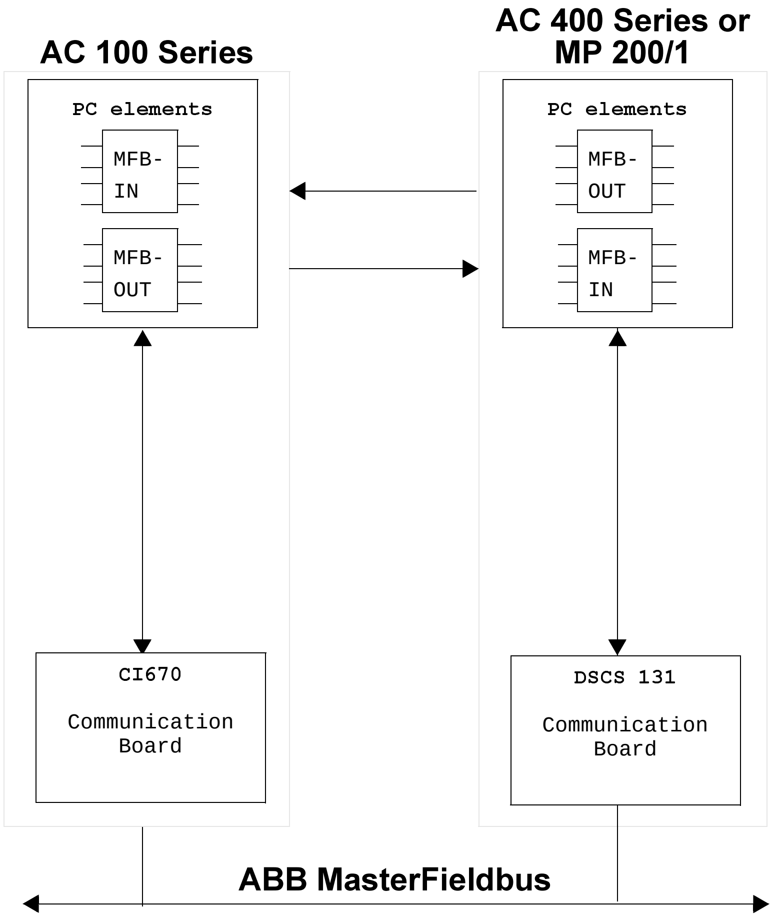
No	Name	Type	Description
F1	UNIT_NO	FI	Number of the unit. Permitted values are 100-115, 200-215, 300-315, 400-415 i.e. 100xbus+first used MFB node -1.
F2	IDENT	FI	IDENTification. Range 1..16. Used to select node area of the MFb.
1	ERR	OB	ERRor. Indicate error at initiating at communication error.
2	ERRTYPE	OI	ERRor TYPE. Error code.
10	PB1	OIL	Packed boolean 1. 32 booleans packed as a 32 bit integer.
: 9+C1	PBC1	OIL	Packed boolean C1. 32 booleans packed as a 32 bits integer.
20	IL1	OIL	Integer Long 1 from the Advant Controller 110.
19+C2	ILC2	OIL	Inter Long C2 from the Advant Controller 110.
30	R1	OR	Real 1 from the Advant Controller 110.
: 29+C3	RC3	OR	Real C3 from the Advant Controller 110.
40	T1	OT	Time 1 from the Advant Controller 110.
39+C4	TC4	OT	Time 4 from the Advant Controller 110.
50	TR1	OTR	TimeReal 1 from the Advant Controller 110.
: 49+C5	TRC5	OTR	TimeReal C5 from the Advant Controller 110.
60	I1	OI	Integer 1 from the Advant Controller 110.
: 59+C6	IC6	OI	Integer C6 from the Advant Controller 110.

Function

PC element MFB-IN is used to transfer values to the Advant Controller 110 from the Advant Controller 400 Series or MasterPiece 200/1. With help of call parameters the number and type of values are chosen. Up to 19 values, depending on data type, can be transferred in one element.

General

On the Advant Controller 110 side the communication board CI670 is used, on Advant Controller 400 Series or MasterPiece 200/1 side the communication board DSCS 130/ DSCS 131. Data exchange between Advant Controller 100 Series and Advant Controller 400 Series or MasterPiece 200/1 is maintained by the PC elements MFB-OUT and MFB-IN. For each MFB unit one pair of these two elements are built in each system.



Communication via MFB-IN/ MFB-OUT

Configuration

The capacity of a MFB-IN (or MFB-OUT) element is 38 bytes. Any combination (within the limit of 38 bytes) of the following data types is possible.

(0...9)×4 bytes for packed Booleans

(0...9)×4 bytes for Integers Long

(0...9)×4 bytes for Reals

(0...9)×4 bytes for Times

(0...9)×4 bytes for Time Reals

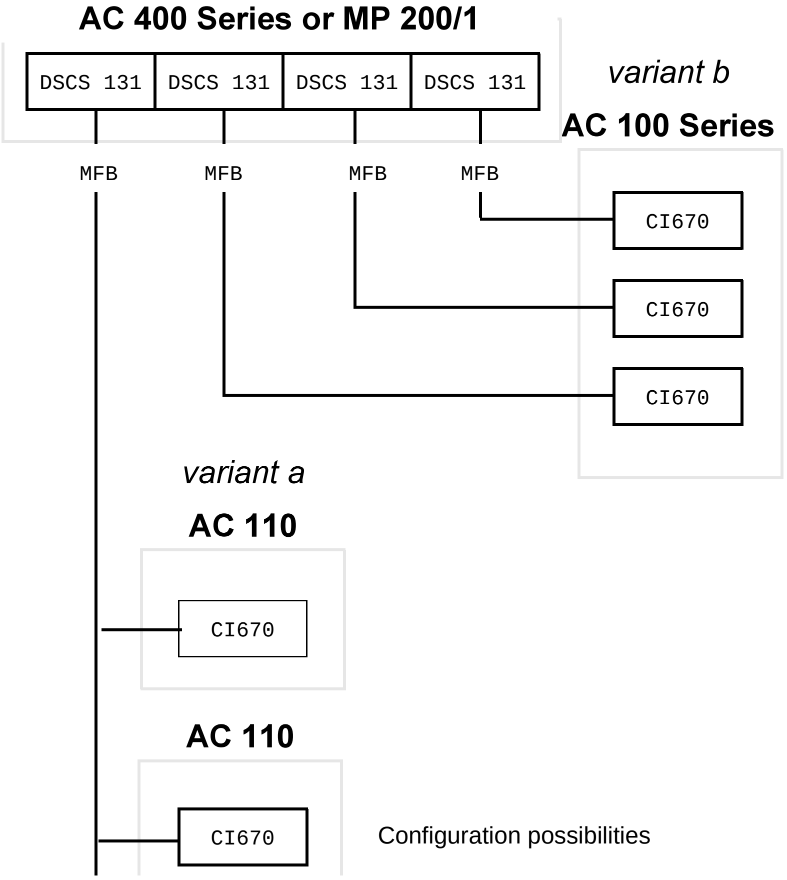
(0...19)×2 bytes for Integers

One pair of MFB-IN / MFB-OUT PC elements occupies one unit number on the MasterFieldbus. One CI670 can use up to 16 unit numbers. The maximum number of units on a MasterFieldbus is 16.

The following two basic configurations are possible:

variant A; one or several Advant Controller 110 on the same MFB

variant B; one Advant Controller 110 connected to one or several MFBs



Configuration possibilities

Address Selection

One Advant Controller 100 Series can occupy up to 16 unit numbers of one MasterFieldbus (MFb). With the function parameter UNIT_NO the first unit number is selected, the function parameter IDENT is used to identify the following unit numbers occupied by the same Advant Controller 100 Series.

If an Advant Controller 100 Series occupies two unit numbers on the MFb, it is possible to use two MFB-IN elements to pick up information from the same Advant Controller 400 Series or MasterPiece 200/1. If the first used unit number is 1, the addressing parameters at the element are set to UNIT_NO=x00 and IDENT=1. The second MFB-IN element shall be addressed to UNIT_NO=x00 and IDENT=2.

Remark: There is a difference in use of the terminal IDENT of PC element MFB-IN between AC 100 Series and AC 400 Series or MP 200/1. In AC 110 the terminal IDENT requires the absolute node number +1, which is equal to the used channel number CI670. E.g. if you use CI670 channels 4 and 5 (UNIT__NO=x03) you have to set IDENT=4 and IDENT=5 at the two MFB-IN in AC 110. In AC 400 Series, IDENT requires a relative nod number. The first node used for communication with every partner has IDENT=1.

Configuration Check

Each minute the call parameters of the MFB-IN element are checked against the call parameters of the MFB-OUT element in AC 400 Series or MP 200/1. When the MasterFieldbus communication has been started it may last up to a minute until data are transferred for the first time.

The call parameters of the MFB-IN element must also be used in the CI670 DB element as CI parameters.

Fault Handling

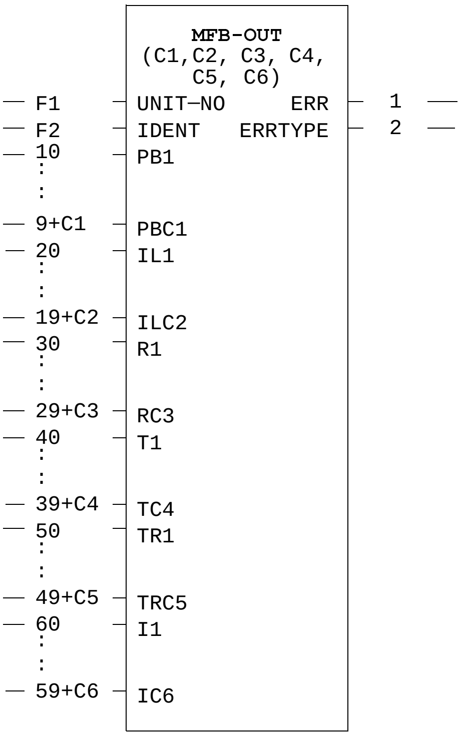
If an error is detected during initialization or communication the output ERR is set to TRUE. The error type is shown in output ERRTYPE. If several errors exist simultaneously, the error code that occurs first is shown and the next remaining error is shown when the first one disappears. Possible error codes are:

ERRTYPE	Error description
0	OK.
-1	The data base element given by the parameter UNIT_NO is missing.
-3	The data base element given by the parameter UNIT_NO is not accepted. This may be caused by an erroneous bus number.
11	The PC element is executing in Reset Mode.
20	The received data is not valid.
-21	The IDENT is smaller than the first UNIT_NO.
22	Communication board CI670 error.
-23	Parameter inconsistency between PC element and DB element.

Output to MasterFieldbus MFB-OUT

Summary

The element MFB-OUT is used for transferring data from Advant Controller 110 to an Advant Controller 400 Series or a MasterPiece 200/1 connected via MasterFieldbus.



PC element MFB-OUT

Call MFB-OUT (C1, C2, C3, C4, C5, C6)

Parameter	Significance	Permissible values
C1	Number of packed booleans	0..9
C2	Number of IL values	0..9
C3	Number of Real values	0..9
C4	Number of Time values	0..9
C5	Number of TimeReal values	0..9
C6	Number of I values	0..19
$2\times C1+2\times C2+2\times C3+2\times C4+2\times C5+C6\leq 19$		

Terminal Description

No	Name	Type	Description
F1	UNIT_NO	FI	Number of the unit. Permitted values are 100-115, 200-215, 300-315, 400-415 i.e. 100xbus+first used MFB node -1.
F2	IDENT	FI	IDENTification. Range 1..16. Used to select node area of the MFb.
1	ERR	OB	ERRor. Indicates error at initiating at communication error.
2	ERRTYPE	OI	ERRor TYPE. Error code.
10	PB1	IIL	Packed boolean 1. 32 booleans packed as a 32 bit integer.
: 9+C1	PB*C1*	IIL	Packed boolean C1. 32 booleans packed as a 32 bits integer.
20	OL1	IIL	Integer Long 1 to the Advant Controller 110.
19+C2	IL*C2*	IIL	Inter Long C2 to the Advant Controller 110.
30	R1	IR	Real 1 to the Advant Controller 110.
: 29+C3	R*C3*	IR	Real C3 to the Advant Controller 110.
40	T1	IT	Time 1 to the Advant Controller 110.
39+C4	T*C4*	IT	Time C4 to the Advant Controller 110.
50	TR1	ITR	TimeReal 1 to the Advant Controller 110.
49+C5	T*C5*	ITR	TimeReal C5 to the Advant Controller 110.
: 59+C6	I*C6*	OI	Integer C6 to the Advant Controller 110.
90	STATUS	IB	Status information from the AC 110 system to the system status function in MV 800 and the PX data base element in AC 400 Series or MP 200/1. OK=1. STATUS=0 leads to error message ”local network failed” at the PX DB element. Only the STATUS input of the first configured node is used.

Function

PC element MFB-OUT is used to transfer values from the Advant Controller 100 Series system to the Advant Controller 400 Series or MasterPiece 200/1 system. With help of call parameters, the number and type of values are selected. Up to 19 values, depending on data type, can be transferred in one element. The available data type is 32 bit packed Booleans, 32 bit Integer, 32 bit Real, 32 bit Time, 32 bit TimeReal and 16 bit Integer.

General and Configuration

See PC element MFB-IN General description and Configuration description, respectively.

Address Selection

One Advant Controller 100 Series can occupy up to 16 unit numbers of one MFb. With function parameter UNIT_NO the first occupied unit number is selected, the function parameter IDENT is used to identify the following unit number. If an Advant Controller 100 Series occupies two unit numbers at the MFb, it is possible to use two MFB-OUT elements to send information to the Advant Controller 400 Series or MasterPiece 200/1 system. If the first unit number is 1, the addressing parameters at the element are set to UNIT_NO=x00 and IDENT=1. The second MFB-IN element shall be addressed to UNIT_NO=x00 and IDENT=2.

Remark: There is a difference in use of the terminal IDENT of PC element MFB-OUT between AC 100 Series and AC 400 Series or MP 200/1. In AC 110 the terminal IDENT requires the absolute node number +1, which is equal to the used channel number CI670. E.g. if you use CI670 channels 4 and 5 (UNIT__NO=x03) you have to set IDENT=4 and IDENT=5 at the two MFB-OUT in AC 100 Series. In AC 400 Series, IDENT requires a relative nod number. The first node used for communication with every partner has IDENT=1.

Configuration Check

Each minute the call parameters of the MFB-OUT element are checked against the call parameters of the MFB-IN element in AC 400 Series or MP 200/1. When the MasterFieldbus communication has been started, it may last up to a minute until data are transferred. The call parameters of the MFB-OUT element must also be used in the CI670 DB element as CO parameters.

Fault Handling

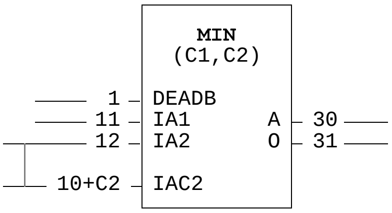
If an error is detected during initialization or communication the output ERR is set to TRUE. The error type is shown in output ERRTYPE. If several errors simultaneously, the error code that occurs first is shown and the next remaining error is shown when the first one disappears. Possible error codes are:

ERRTYPE	Error description
0	OK.
-1	The data base element given by the parameter UNIT_NO is missing.
-3	The data base element given by the parameter UNIT_NO is not accepted. This may be caused by an erroneous bus number.
11	The PC element is executing in Reset Mode.
-21	The IDENT is smaller than the first UNIT_NO.
22	Communication board CI670 error.
-23	Parameter inconsistency between PC element and DB element.

Minimum Selector MIN

Summary

MIN (**MIN**imum selector) is used to select the lowest value of an optional number of integers or real numbers.



PC element MIN

Call MIN (C1, C2)

Parameter	Significance	Permissible values
C1	Data Type	I, IL, R
C2	Number of input	2..19

Terminal Description

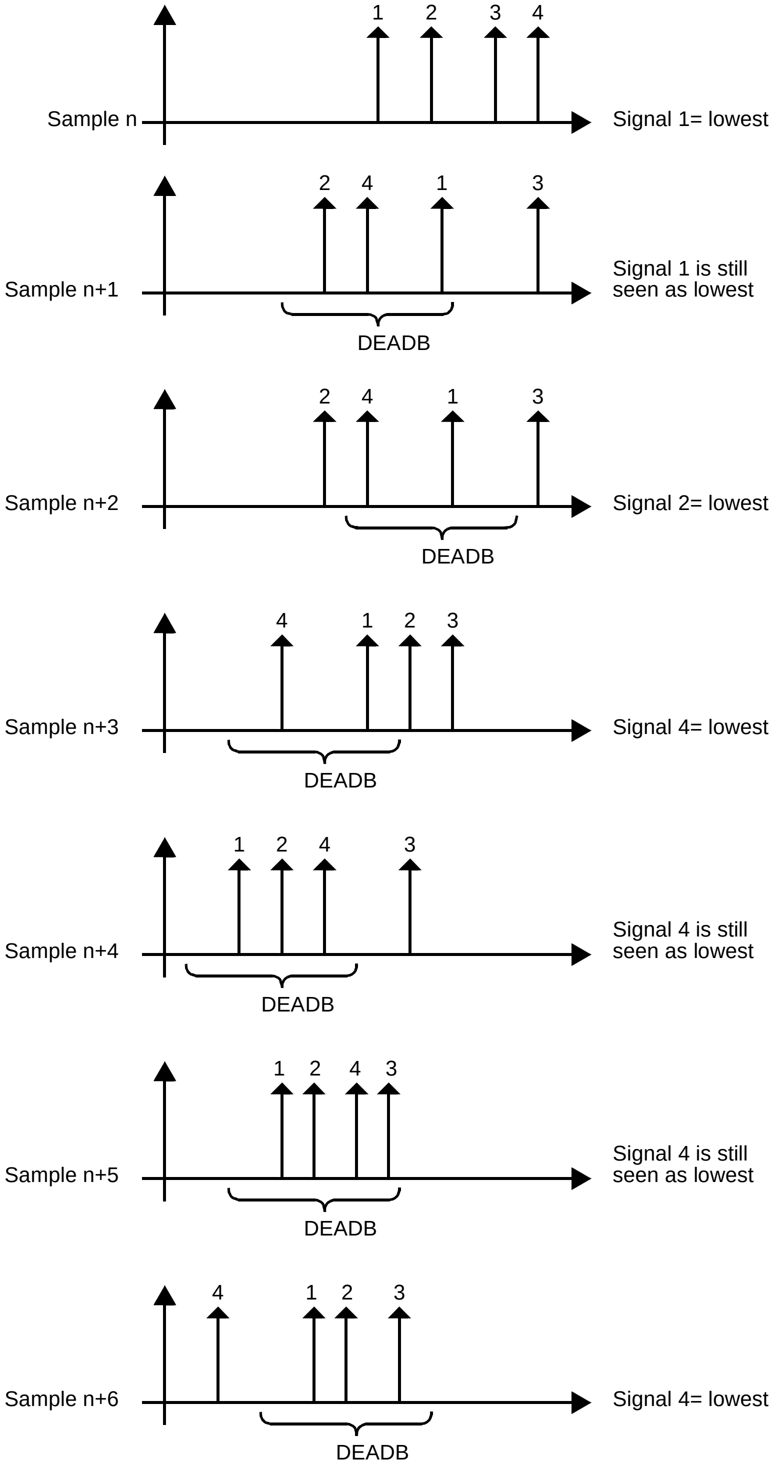
No	Name	Type	Description
1	DEADB	I*C1*	**DEADB**and. Input for deadband.
11	IA1	I*C1*	**I**nput **A**ddress **1**. Input the value of which is compared with the other inputs.
12	IA2	I*C1*	**I**nput **A**ddress **2**. Input the value of which is compared with the other inputs.
: 10+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. Input the value of which is compared with the other inputs.
30	A	OI	**A**ddress. Output for the number of the input having the lowest value.
31	O	I*C1*	**O**utput. Output for the least value.

Function

The values at the inputs IA1 … IA*C2* are compared and the lowest value is obtained at the output O. The number of the input with the lowest value is obtained at the output A. If the two smallest signal values are equal when the element is executed the first time, the signal with the lowest connection number is selected as the smallest.

Deadband

The deadband specified at the input DEADB is symmetrical around the value of the least input. The upper and lower deadband limits are calculated from the value for the lowest input in the preceding sample. To prevent rapid changes at the output A, the value at A is retained until the value at the corresponding input exceeds the calculated upper deadband limit or until one of the other inputs falls below the lower deadband limit.

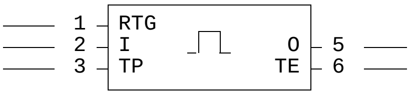


Example of function with deadband

Monostable MONO

Summary

The mono function MONO can be used, for example, for time limiting of operation of outputs and automatic functions. The element can also be used for impulse extension, stall alarm function, etc.



PC element MONO

Call MONO

Terminal Description

No	Name	Type	Description
1	RTG	IB	**R**e**T**rig**G**. Input for selection of if the MONO function is to be retriggerable or not. If RTG is set, a retriggerable function is obtained.
2	I	IN	Input. Input for start of the MONO function when the input changes from 0 to 1.
3	TP	IT	**T**ime **P**ulse. Input for setting of pulse time.
5	O	OB	**O**utput. Output which is set when the MONO function starts and resets (to 0) when the time set has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output for time which has elapsed when O is set. When the time set (TP) has elapsed TE will stop.

Function

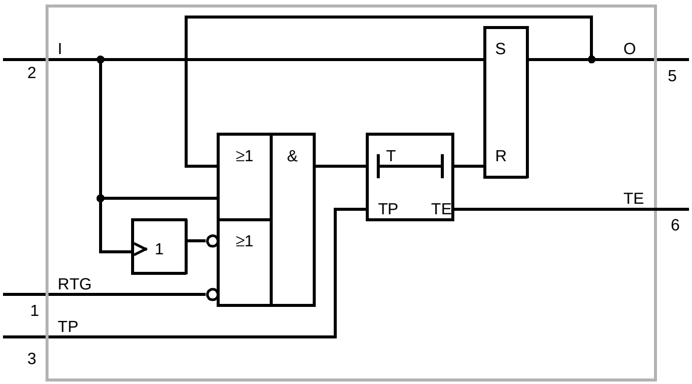
A memory is set when the input I is set. The output then goes to 1, see function diagram. When the time set in the timer has elapsed, the memory is cleared and the output O goes to 0.

MONO Function, Not Retriggerable

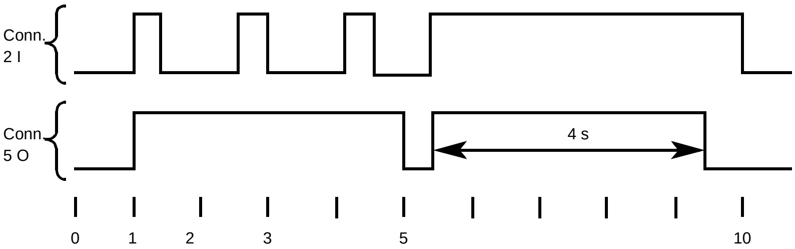
If the input RTG is reset, a MONO function which is not retriggerable is obtained. If a new pulse is obtained at the input I before the time set in the timer has elapsed, it does not affect the timer. Only when the time set has elapsed and the output O is reset can the MONO function be restarted by the input I going from 0 to 1.

MONO Function, Retriggerable

If RTG is set, a retriggerable MONO function is obtained, i.e. the timer starts from 0 each time a new pulse is obtained at the input I. If a new pulse is obtained at the input I before the time set in the timer has elapsed, the timer will restart, i.e. the MONO function is retriggerable.

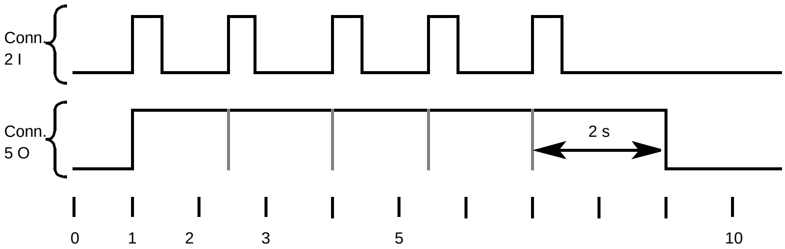


Function diagram



Conn.3 Time set=4 s
Conn.1 Reset

Time diagram MONO function, not retriggerable



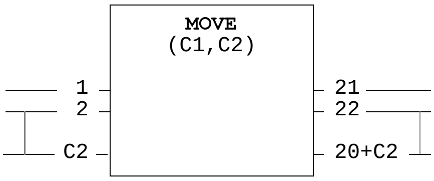
Conn.3 Time set=2 s
Conn.1 Set

Time diagram MONO function, retriggerable

Data Copying MOVE

Summary

MOVE copies the value at the input to the output and may be used to e.g. copy the value from one location in the data base to another.



PC element MOVE

Call MOVE (C1, C2)

Parameter	**Significance**	**Permissible values**
C1	Data type	B, I, IL, R, T, TR
C2	Number of inputs and outputs	1..19

Terminal Description

No	**Name**	**Type**	**Description**
1	–	I*C1*	Input for value 1.
2	–	I*C1*	Input for value 2.
: C2	–	I*C1*	Input for value C2.
21	–	O*C1*	Output for value 1.
22	–	O*C1*	Output for value 2.
: 20+C2	–	O*C1*	Output for value C2.

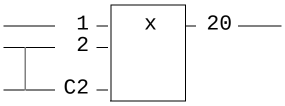
Function

The element copies the values at the inputs to the respective output. The type of the data to be moved is determined with the call parameter C1 and the number of values moved is determined with the call parameter C2.

Multiplier MUL

Summary

MUL (**MUL**tiplier) is used for multiplication of an optional number of integers or real numbers.



PC element MUL

Call MUL (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2..19

Terminal Description

No	Name	Type	Description
1	I*C1*		Input for multiplicand
2	I*C1*		Input for multiplier
: C2	I*C1*		Input for multiplier
20	O*C1*		Output for product

Function

The values at the inputs 1 … C2 are multiplied and the product is stored at the output 20.

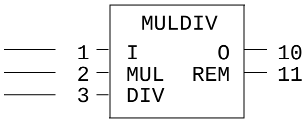
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Integer scaling element MULDIV

Summary

MULtiplier **DIV**ider is used to scale integer values by dividing the product of two integer inputs.



PC element MULDIV

Call MULDIV

Terminal Description

No	Name	Type	Description
1	I	II	**I**nput for multiplicand
2	MUL	II	Input for **MUL**tiplier
3	DIV	II	Input for **DIV**isor
10	O	OI	**O**utput for quotient
11	REM	OI	Output for **REM**inder

Function

The product of input I and input MUL is divided by input DIV. The quotient is loaded at the output O and the reminder at REM. The element internally uses 32 bit accuracy to perform the multiplication.

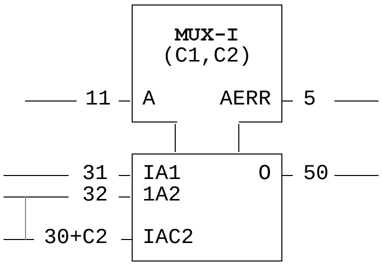
Overflow

If the maximum positive value is exceeded, then the output O is limited to +32767. If the minimum negative value is exceeded, then the output O is limited to -32767

Multiplexer with Integer Address MUX-I

Summary

MUX-I (**MU**ltiple**X**er **-** with **I**nteger address) is used as a selector and has an optional number of inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-I

Call MUX-I (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	1..19

Terminal Description

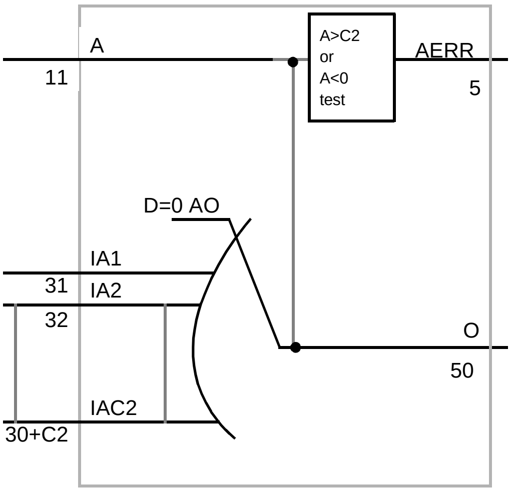
No	Name	Type	Description
5	AERR	OB	**A**ddress **ERR**or. Output which is set when the address value is larger than the number of inputs, or negative.
11	A	II	**A**ddress. Input for address value which specifies which input is to be connected to the output O. When A=0, the output is set to 0.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The second input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. The last input to the selector.
50	O	O*C1*	**O**utput. Data output from the selector.

Function, Addressing

The input, the data value (IA1 … IA*C2*) of which is to be connected to the output O, is specified with an address (integer 1 … C2) at the input A. If the address is 0, the output terminal O is set to zero.

Supervision

The address A is monitored and if its value is greater than the number of inputs or is negative, the error signal output AERR is set. The data value 0 is then obtained at the output.

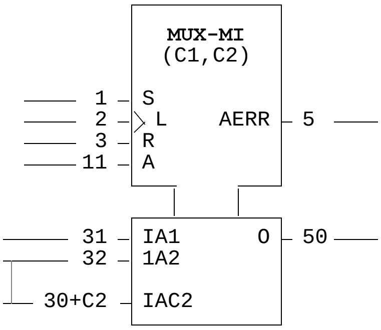


Function diagram

Multiplexer with Memory and Integer Address MUX-MI

Summary

MUX-MI (**MU**ltiple**X**er **-** with **M**emory and **I**nteger address) is used as a selector with memory function and has up to 19 inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-MI

Call MUX-MI (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	1..19

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading of new values each time the element is executed. When the input S is reset 0, the last value loaded will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of the data addressed.
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	AERR	OB	**A**ddress **ERR**or. Output which is set when the address value is greater than the number of inputs or is negative.
11	A	II	**A**ddress. Input for address value which specifies from which input data is to be loaded. When A=0, the output is reset to 0.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The first input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	A*C2*	I*C1*	**I**nput **A**ddress ***C2***. The last input to the selector.
50	O	O*C1*	**O**utput. Data output from the selector.

Function, Addressing

The input data value (IA1 … IA*C2*) that is to be stored at the output is specified with an address (integer 1 … C2) at the input A. If the address is 0, the data value 0 is stored.

Loading

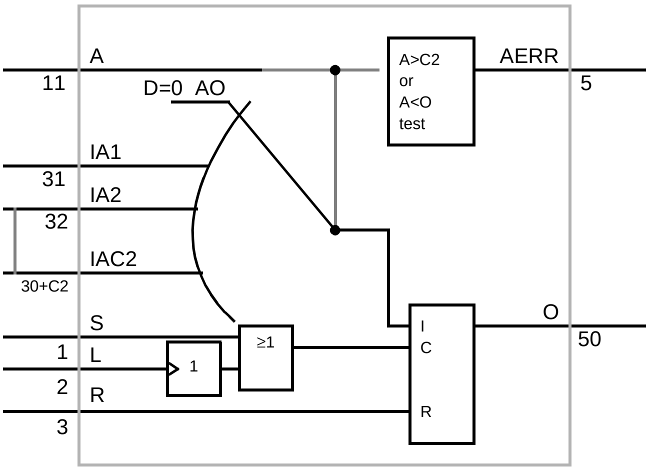
The value is loaded at the instant the input L is set to 1. If input S is set, new data is loaded each time the element is executed. When S is reset after having been set, the data loaded most recently remains until the element is once more executed with input S, L or R set. Input S overrides input L, i.e. when S is set, L has no function.

Clearing

The input R clears the data output and prevents further storage of data.

Supervision

The address A is monitored and if its value is greater than the number of inputs or is negative, the error signal output AERR is set. The data value 0 is then stored at the output.

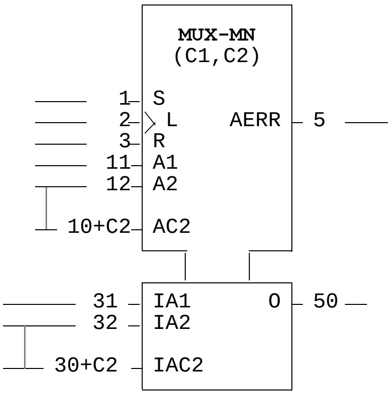


Function diagram

Multiplexer with Memory and 1 of N Address MUX-MN

Summary

MUX-MN (**MU**ltiple**X**er **-** with **M**emory and 1-of-**N** address) is used as a selector with memory function. MUX-MN has up to 19 inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-MN

Call MUX-MN (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	1..19

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading of new values each time the element is executed. When the input S is reset, the last value loaded will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of the data addressed.
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	AERR	OB	**A**ddress **ERR**or. Output which is set when two or more of the inputs A1 .. A*C2* are set.
11	A1	IB	**A**ddress **1**. Input to store data from the input IA1 at the output O.
12	A2	IB	**A**ddress **2**. Input to store data from the input IA1 at the output O.
10+C2	A*C2*	IB	**A**ddress ***C2***. Input to store data from the input IA*C2* at the output O.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The first input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	IA*C2*	I*C1*	**I**nput **A**ddress *C2*. The last input to the selector.
50	O	OC1	**O**utput. Data output from the selector.

Function, Addressing

The input data value (A1 … IA*C2*) that is to be stored at the output is specified with the inputs A1 … A*C2*. If the input A1 is 1, the value from input IA1 is stored at the output, if A2 is 1, the value from input IA2, etc. If none of the inputs A1 … A*C2* is set to 1, the data value 0 is stored at the output O.

Loading

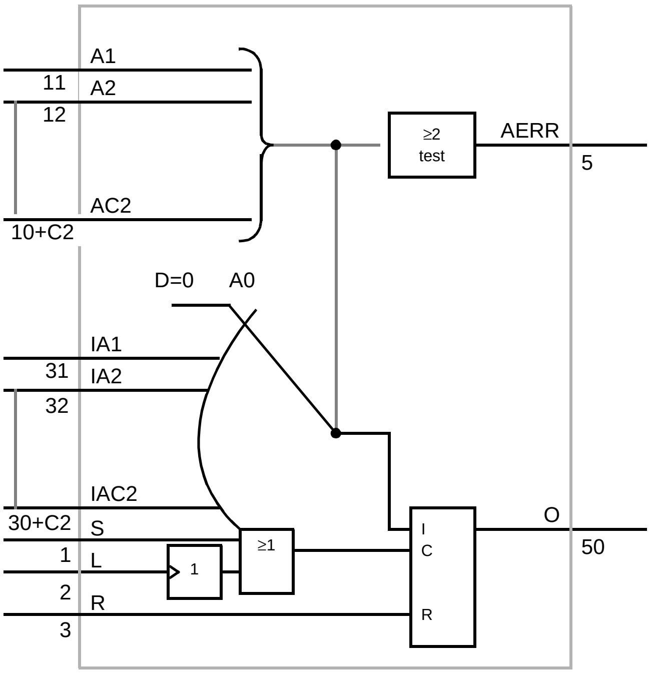
The value is loaded at the instant the input L is set to 1. If input S is set, new data is loaded each time the element is executed. When S is reset after having been set, the data loaded most recently remains. Input S overrides input L, i.e. when S is set, L has no function.

Clearing

The input R clears the data output and prevents further storage of data.

Supervision

If two or more of the inputs A1 … A*C2* are set simultaneously, the output AERR is set and the value from the input which correspond to the address input set with the lowest number is stored at the output O.

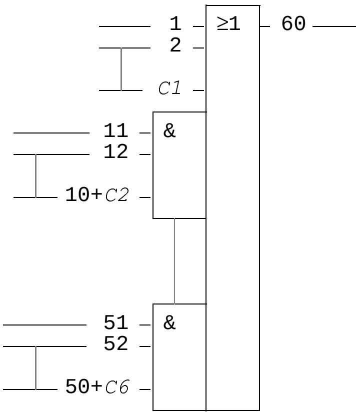


Function diagram

Or Gate with And Inputs OR-A

Summary

OR-A (**OR -** with **A**nd gates on the inputs) is used to form general combined expressions with Boolean variables.



PC element OR-A

Call OR-A (C1, C2, C3, ... Cn)

Parameter	Significance	Permissible values
C1	Number of inputs in the OR part of the element	0. 9
C2	Number of inputs in the first AND gate of the element	0..9
C3	Number of inputs in the second AND gate of the element	0..9
: C*n*(1)	Number of inputs in the last AND gate of the element	0..9

(1)1≤*n*≤6
1≤C1+C2+...+C*n*≤40

Terminal Description

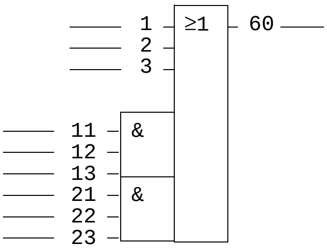
No	Name	Type	Description
1	—	IB	Input direct to the OR gate in the element.
2	—	IB	Input direct to the OR gate in the element.
: *C1*	—	IB	Input direct to the OR gate in the element.
11	—	IB	Input to the first AND gate in the element.
12	—	IB	Input to the first AND gate in the element.

Terminal Description

No	Name	Type	Description
10+*C2*	—	IB	Input to the first AND gate in the element.
21	—	IB	Input to the second AND gate in the element.
22	—	IB	Input to the second AND gate in the element.
: 20+*C3*	—	IB	Input to the second AND gate in the element.
10x(*n*-1)+1	—	IB	Input to the last AND gate in the element.
10x(*n*-1)+2	—	IB	Input to the last AND gate in the element.
: 10x(*n*-1)+C*n*	—	IB	Input to the last AND gate in the element.
60	—	OB	Output.

Function

More rapid execution than if separate elements are used can be obtained by using assembled elements such as OR-A. The inputs need then only be tested until the value of the output can be determined. If input 1 is 1, the output can be set irrespective of the other inputs. See the figure and the table below.



PC element OR-A (3, 3, 3)

Table OR-A (3, 3, 3)

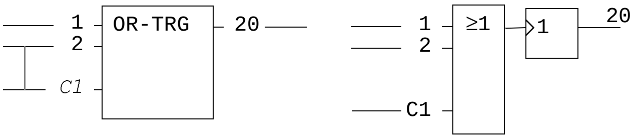
1	**2**	**3**	**11**	**12**	**13**	**21**	**22**	**23**	**60**
0	0	0	0	0	0	1	1	1	1
0	0	0	0	x	x	0	x	x	0
0	0	0	0	x	x	1	0	x	0
0	0	0	0	x	x	1	1	0	0
0	0	0	1	0	x	0	x	x	0
0	0	0	1	0	x	1	0	x	0
0	0	0	1	0	x	1	1	0	0
0	0	0	1	1	0	0	x	x	0
0	0	0	1	1	0	x	0	x	0
0	0	0	1	1	0	x	x	0	0
0	0	0	1	1	1	x	x	x	1
0	0	1	x	x	x	x	x	x	1
0	1	x	x	x	x	x	x	x	1
1	x	x	x	x	x	x	x	x	1

x indicates that the input has no effect on the value of the output or the execution time.

Or Gate With Trigger on Outputs OR-TRG

Summary

OR-TRG (**OR** - with **TRiG**ger on the output) is used when the output of an OR gate should be triggered.



Call AND-TRG (C1)

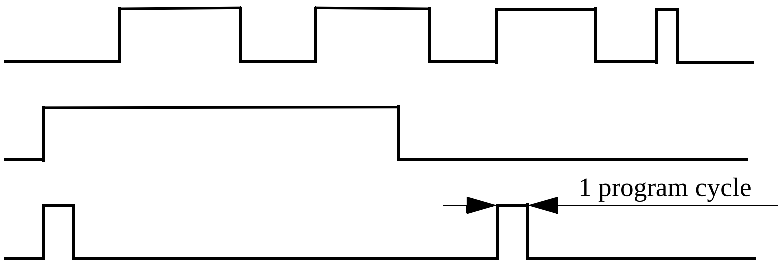
Parameter	Significance	Permissible values
C1	Number of OR inputs	2..19

Terminal Description

No	Name	Type	Description
1	--	IB	Input to the OR gate.
2	--	IB	Input to the OR gate.
: *C1*	--	IB	Input to the OR gate.
20	--	OB	Output

Function

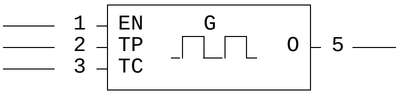
When the result of the OR gate is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the result of the OR gate. For the output signal to be set, the result of OR gate must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.



Oscillator OSC-B

Summary

Oscillator OSC-B (**OSC**illator **- B**oolean variables) with variable frequency and pulse time is used when pulse trains with periods from 2 × cycle time up to 24 hours are needed.



PC element OSC-B

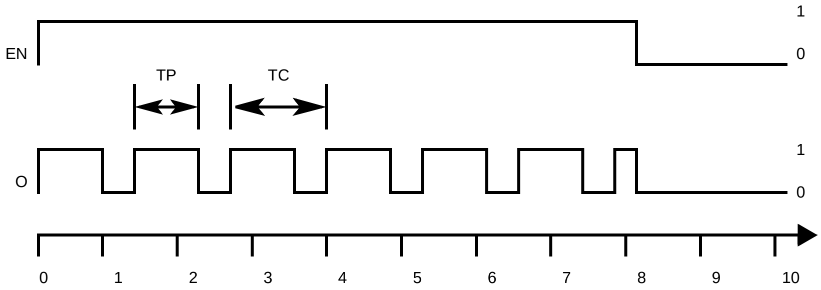
Call OSC-B

Terminal Description

No	Name	Type	Description
1	EN	IB	**EN**able. Input for start of oscillator.
2	TP	IT	**T**ime **P**ulse. Input for setting of pulse time.
3	TC	IT	**T**ime **C**ycle. Input for setting of period. The period must be greater than the pulse duration.
5	O	OB	**O**utput. Oscillator output.

Function

The oscillator starts when the input EN is set (to 1). The figure below shows an example of an oscillator time diagram.

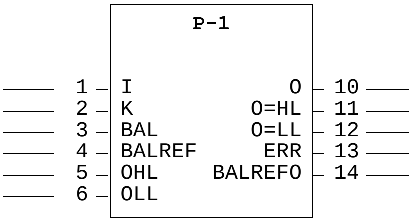


Example of time diagram with TP=1 s and TC=1.5 s

Proportional Regulator P-1

Summary

P-1 (**P**roportional Function) is used as an amplifier in feedback systems. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference.



PC element P-1

Call P-1

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input signal.
2	K	IR	Input for setting gain.
3	BAL	IB	**BAL**ance. Input for activation of following.
4	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
5	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
6	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.
14	BALREFO	OR	**BAL**ance **REF**erence **O**utput. Information output for O/K.

Function

The P-function can be written in the time plane as O(t)=K I(t). Its main property when controlling is that the output signal is directly proportional to the input signal. The transfer function for a P-function is: G(S)=K

Following

If BAL is set to 1, the amplifier goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned.

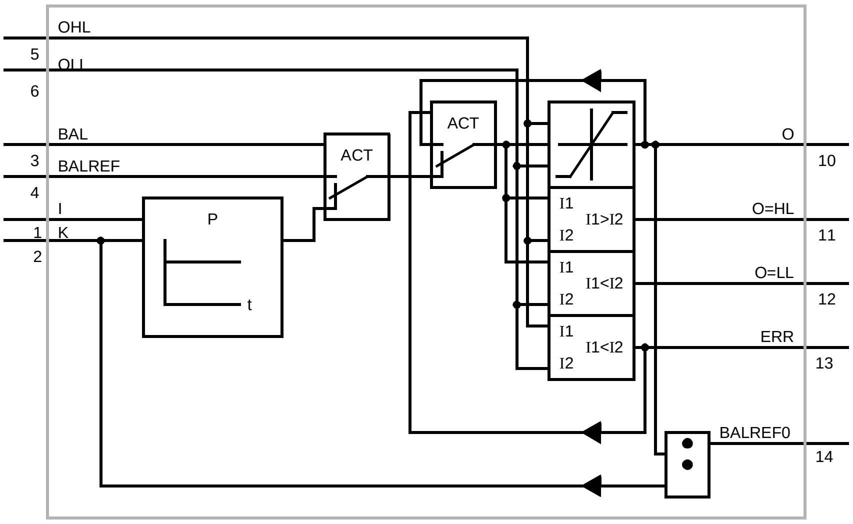
Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

The element checks that the upper limit OHL is greater than the lower limit OLL. If not, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred.

Information Output BALREFO

The output BALREFO gives the result of the calculation O(t)/K where O(t) is the output signal after the limitation. The output specifies the input signal which corresponds to the output signal concerned.

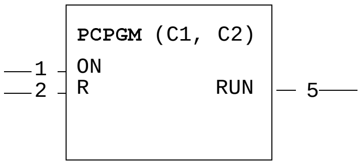


Function diagram

Program Header PCPGM

Summary

The program header PCPGM is used for the structuring and execution control of the application program of a PC system.



PC element PCPGM

Call PCPGM (C1, C2)

C1	Cyclicity in ms	Allowed values are between 2ms and 20000ms in steps of 2ms
C2	Place in the cycle time table and/or scheduling strategy	1..251: place in cycle time table, cyclic activation 252: activation at initialization after power down.

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) with normal execution.
2	R	IB	**R**eset. Clearing of the execution units of the PC program.
5	RUN	OB	**RUN** indicates that the execution units in the PC program may be executed.

Function

A PC program is the highest level in the PC program structure. The program is intended to perform a complete control function. The program header PCPGM is used to control the enabling, disabling and resetting of the complete PC program, irrespective of subordinate execution units. The call parameter C1 specifies how often the program header is to be executed and the call parameter C2 is used to specify the position in the cycle time table on which the execution order of the execution units with the same cycle time is based.

PCPGM must be given a cyclicity from the table for the interpreter with the highest priority. This is required for changes at the inputs to PCPGM to have a direct effect on subordinate execution units. The program header cannot have directly subordinate elements. The control inputs ON and R are used to control when and how the complete PC program with subordinate execution units is to be executed. PCPGM itself is always executed, irrespective of the inputs ON and R.

Normal Execution

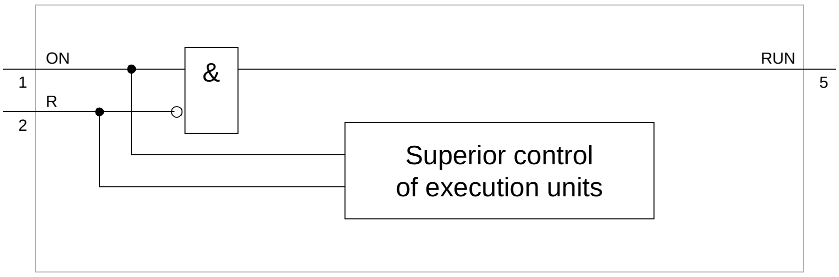
The input ON must be set (to 1) for the normal execution of subordinate execution units to be possible. At the start of the execution of subordinate execution units, a check is performed to ensure that the program header has RUN status.

Clearing

If the input R is set, the complete PC program is executed in reset mode. This means that all outputs on the elements in the slave modules are given initial values, which in most cases are the 0-value of the data concerned. Reading to the data base is performed with data value 0 for all data types. Input R overrides input ON.

RUN

Output RUN is set only if normal execution is permissible, i.e. if the ON input is set and the R input is reset (to 0).

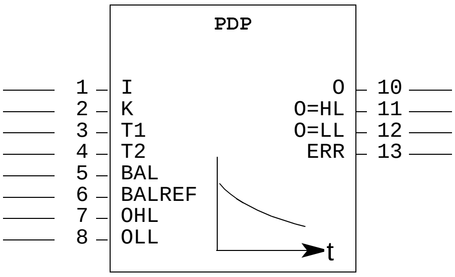


Function diagram

PDP-Function PDP

Summary

PDP (**P**roportional **D**erivating **P**roportional function) is used to give proportional effect and limited derivation effect.
The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from balancing or from limited output signal are bumpless.



PC element PDP

Call PDP

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T1	ITR	**T**ime **1**. Input for time constant for lead.
4	T2	ITR	**T**ime **2**. Input for time constant for lag.
5	BAL	IB	**BAL**ance. Input for activation of following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

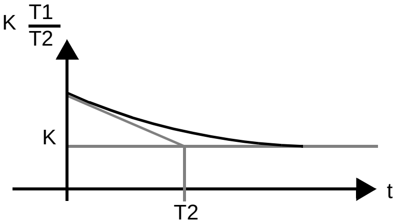
Function

The step response for a PDP function is:

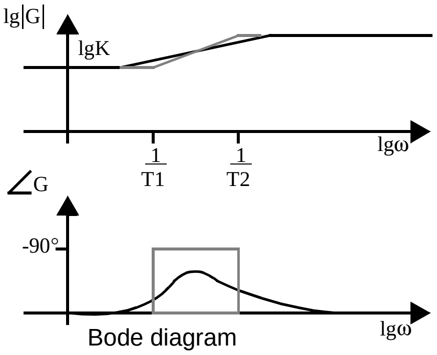
$O(t)=I(t)\ (K+k1e^{-t/T2})$ where $k1=K(T1/T2-1)$

The transfer function for a PDP function is:

$G(s)=K(1+sT1)/(1+sT2)$ where $T2<T1$



Step response

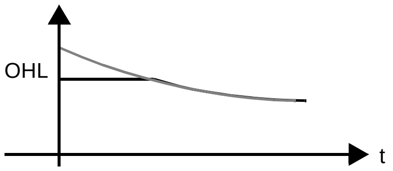


Bode diagram

This has been implemented in the PDP element as a recursive algorithm. The design of this algorithm is such that normal functioning is maintained even during limiting. This ensures a controlled return to a dynamic state.

Gain, Time Constants and Sampling Time

Certain constants are pre-calculated to make the execution time of the element as short as possible. The results are stored internally in the element. These constants are recalculated if T1, T2 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check whether T1 and $T2 \geq 2 \times TS$. If this is not the case T1 or T2 respectively is set equal to $2 \times TS$.



Function in limiting state

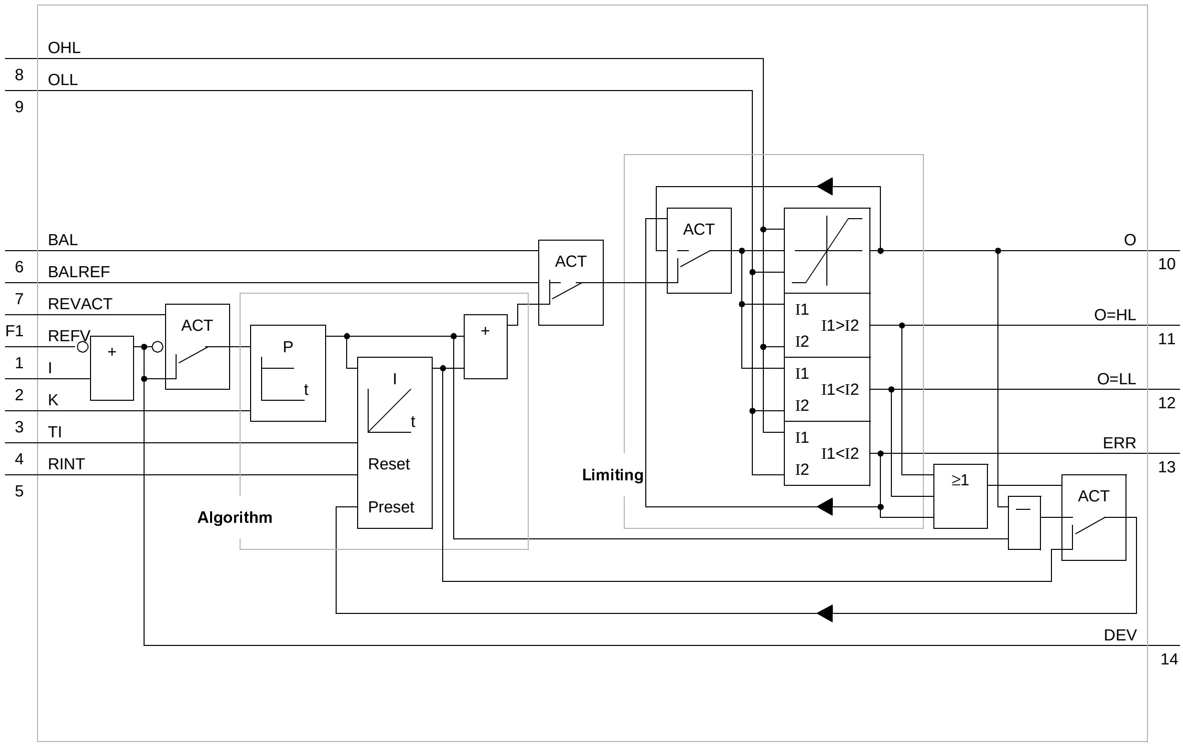
Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limits value concerned. Return to dynamic state is bumpless.

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for the upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the value they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following.

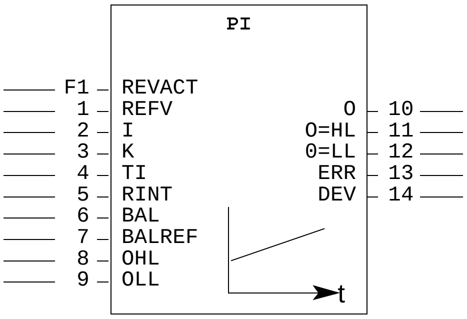


Function diagram

PI-Regulator PI

Summary

PI (**P**roportional **I**ntegrating regulator) is used as a standard PI- regulator for serial compensation in feed back systems. The control deviation is calculated internally in the element. The output signal can be limited to limits specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to normal function. All transfers from balancing or limited output signal are bumpless.



PC element PI

Call PI

Terminal Description

No	Name	Type	Description
F1	REVACT	IB	**REV**erse **ACT**ion. Parameter input for inverted control action. REVACT=1 gives inverted control action.
1	REFV	IR	**REF**erence **V**alue. Input for desired value.
2	I	IR	**I**nput. Input for actual value.
3	K	IR	Input for setting gain.
4	TI	ITR	**T**ime **I**ntegration. Input for time constant for integration.
5	RINT	IB	**R**eset **INT**egrator. Input for clearing of integrator.
6	BAL	IB	**BAL**ance. Input for activation of following.
7	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
8	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit.
9	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.
14	DEV	OR	**DEV**iation. Output for control deviations, i.e. DEV=1–REFV.

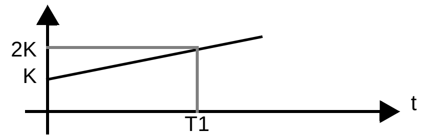
Function, Control Deviation

DEV, the control deviation, is calculated as follows even when the regulator is following: DEV=I–REFV

Inverted Control Effect

REVACT is a function parameter which controls the direction of the output signal change in relation to the actual value.

If REVACT is set to 0, direct action is obtained, i.e. increasing actual value gives increasing output signal. When REVACT is set to 1, inverted action is obtained, i.e. increasing actual value gives decreasing output signal.



Step response

Transfer Function

The PI function can be written in the time plane as

$O(t)=K(DEV(t)+\int^{\tau} DEV(\tau)/TI\ d\ \tau)$

and its main property when controlling is that it permits the integral section to retain its value when the error DEV(t)=0, i.e. the output signal is then constant.

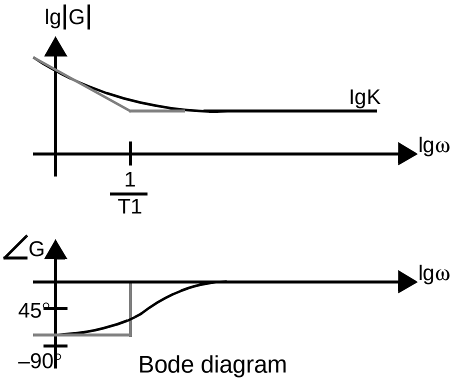
The transfer function for a PI function is:

$G(s)=K(1+1/sTI)$

This has been implemented in the PI element as a recursive algorithm.

Gain, Integration Time Constant and Sampling Time

The constant $K \times TS/TI$ is pre-calculated to make the execution time of the element as short as possible. The result is stored internally in the element. This constant is recalculated if TI or K is changed more than 1/128 of the preceding value or if the sampling time TS is changed. When the value is recalculated a test is performed to ensure that TS/TI<1. If not, TS/TI is set to 1.



Bode diagram

Clearing of Integrator

The integration section of the algorithm is cleared when RINT goes to 1. If the proportional section of the algorithm exceeds the limits of the output signal, the limit status remains and the internal status is updated in accordance with the section "Bumpless transfer from following or limitation". When RINT is set permanently to 1, the element functions as a P-regulator.

Following

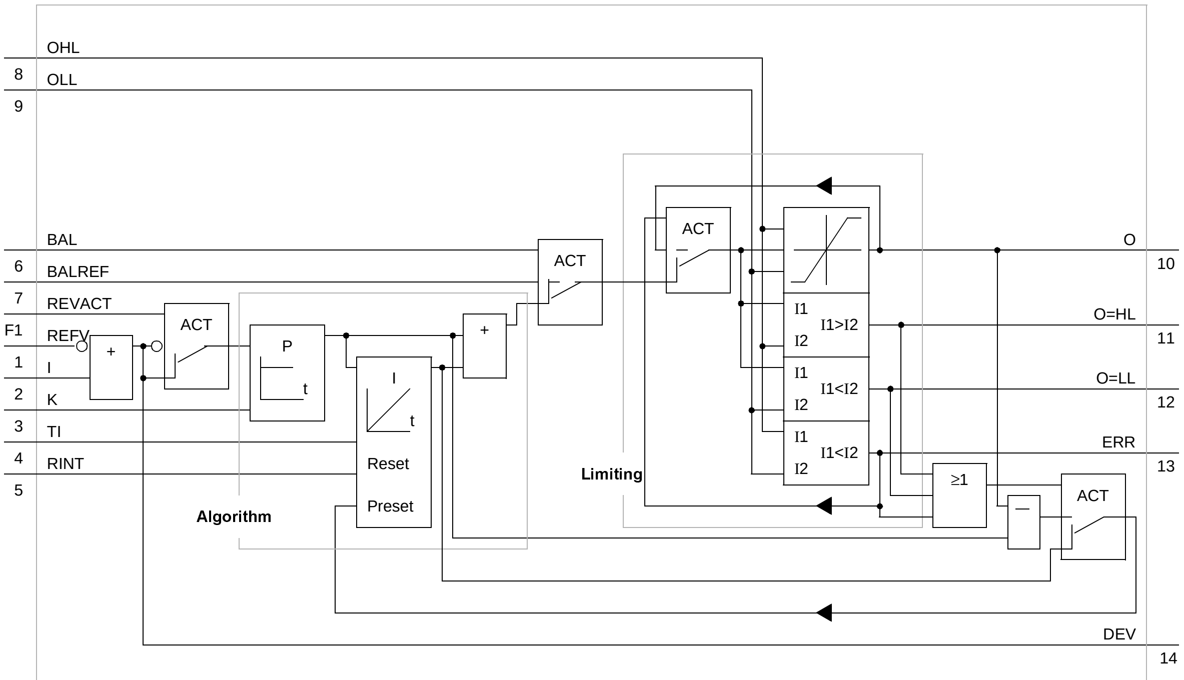
If BAL is set to 1, the regulator immediately goes into following and the output O follows the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. On return to a normal function the value of the output O during the last sample in the following remains during one sample time. See below under "Bumpless transfer from following or limiting".

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit value, the output O=HL is set to 1 and if it falls below the lower limit value, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred.

Bumpless Transfer from Following or Limitation

Transfer from following status (BAL=1) or from a limited output signal is bumpless. This is performed by recalculation of internal states, i.e. the integration part according to $INT(t)=O(t) - K \times DEV(t)$.

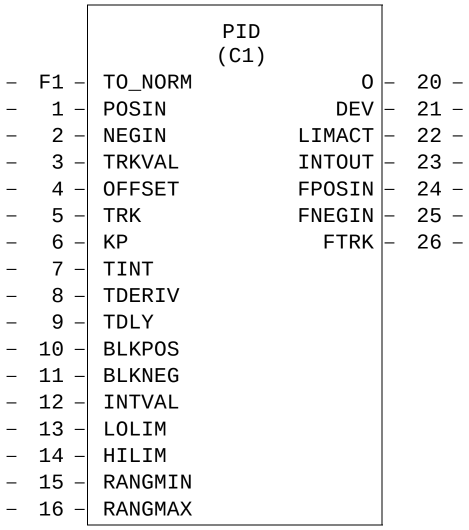


Function diagram

PID algorithm PID

Summary

The element PID (**PID** algorithm) is a PID controller and it is used to create simple control loops on the function group and single-variable level.



PC element PID

Call P_PID (C1)

Parameter	Significance	Permissible values
C1	in AC110 not used	1

Terminal Description

No	Name	Type	Description
F1	TO_NORM	IB	Switch for: Standardization **TO NORM**al output signal. Default value 0.
1	POSIN	IR	**POS**itive **IN**put. *
2	NEGIN	IR	**NEG**ATIVE **IN**put. *
3	TRKVAL	IR	**TR**ac**K**ing **VAL**ue for integrator.
4	OFFSET	IR	**OFFSET**. Disturbance quantity connection.
5	TRK	IB	**TR**ac**K**ing. Input command for MANUAL mode.
6	KP	IB	**P**roportional coefficient. *
7	TINT	IT	**T**ime constant for **INT**egration. *
8	TDERIV	IT	**T**ime constant for **DERIV**ative action. *
9	TDLY	IT	**D**e**L**A**Y** **T**ime. *

Terminal Description

No	Name	Type	Description
10	BLKPOS	IB	Integrator **BL**oc**K**ed for **POS**IN>NEGIN.
11	BLKNEG	IB	Integrator **BL**oc**K**ed for POSIN<**NEG**IN.
12	INTVAL	IR	**INT**egration **VAL**ue. Follow-up control of integrator.
13	LOLIM	IR	**LO**w **LIM**it value of corrective variable. Default value 0.
14	HILIM	IR	**HI**gh **LIM**it value of corrective variable. Default value 100.
15	RANGMIN	IR	Measurement **RANG**e **MIN**imum value.
16	RANGMAX	IR	Measurement **RANG**e **MAX**imum value.
20	O	OR	**O**utput.
21	DEV	OR	**DEV**iation output.
22	LIMACT	OB	Annunciation: **LIM**itation is **ACT**ive.
23	INTOUT	OR	**INT**egrator **OUT**put.
24	FPOSIN	OR	**F**eedback **POS**itive **IN**put.
25	FNEGIN	OR	**F**eedback **NEG**ative **IN**put.
26	FTRK	OR	**F**eedback **TR**ac**K**ing value.

* This value must be entered.

Function

Modes of operation

The element can be operated in either one of 2 operating modes:

- MANUAL mode (balancing mode)
- AUTOMATIC mode (controlled mode)

MANUAL mode

In MANUAL mode, a logic 1 signal is present at input TRK. In this case, controller output O is slaved without delay to the value of the balancing quantity present at input TRKVAL.

The D portion is ineffective in this operating mode.

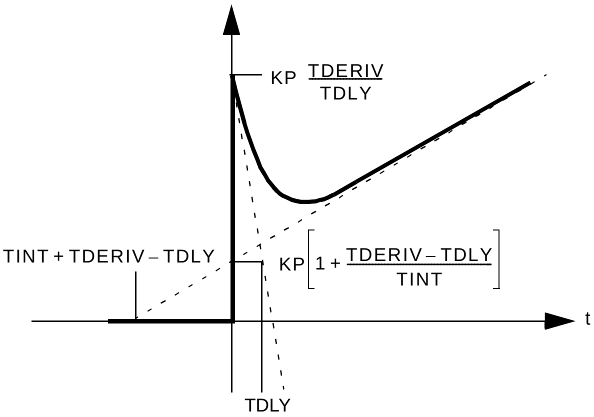
AUTOMATIC mode

In AUTOMATIC mode, a logic 0 signal is present at input TRK.

In this case, the transfer function of the PID controller is:

$$F(s) = KP * \frac{1 + TINT * s}{TINT * s} * \frac{1 + TDERIV * s}{1 + TDLY * s}$$

The transfer function of the PID controller is represented in the following diagram:



The values for proportional coefficient, reset time, derivative action time, and delay time are defined by corresponding entries at inputs KP, TINT, TDERIV and TDLY.

In the case of TDERIV = TDLY, this controller behaves just like a PI controller.

Disturbance quantity connection

At input OFFSET, a disturbance quantity can be connected additionally. The time response of the controller is bypassed, and the disturbance quantity is directly switched through to output O.

Disturbance quantity limitation

Output variable O can be limited to an upper and a lower limit value using inputs HILIM and LOLIM.

When HILIM/LOLIM are effective, the integrator will be controlled accordingly by an internal function of the element, and the D portion will not be considered in that case.

Whenever upper and lower limit should overlap, the upper limit will be given priority.

When a limit value is effective, output O will follow this limit value without delay.

Integrator stop

The I portion of the controller can be inhibited at input BLKPOS by a logic 1 signal in the positive direction (POSIN>NEGIN), and at input BLKNEG by a logic 1 signal in the negative direction (POSIN<NEGIN).

Follow-up control of integrator

When the element is used as a master controller, and if a limit value is effective on a subordinate level, the integrator can be controlled smoothly via input INTVAL.

Input INTVAL can also be used for applying a synchronizing variable for several controllers operating in parallel.

Standardization of output signal

TO_NORM = 0

In this case, TDERIV, TDLY and TINT are related to an adjustment from measurement range RANGMIN to RANGMAX and not to the value 100.

TO_NORM = 1

In this case, input IN is standardized to RANGMIN, RANGMAX.

$$IN' = \frac{IN - RANGMIN}{RANGMAX - RANGMIN} * 100$$

where IN means POSIN and NEGIN.

In this case, POSIN has always be replaced by POSIN’ and NEGIN by NEGIN’ in this document.

Annunciation functions

Output DEV

The discrepancy between inputs NEGIN and POSIN will be put out at deviation output DEV (DEV = NEGIN - POSIN).

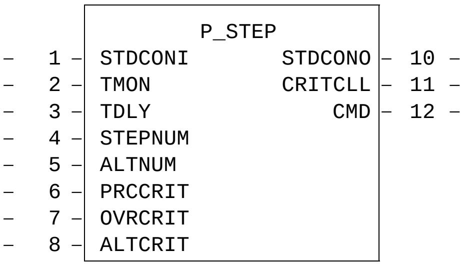
Output LIMACT

When the upper or lower limit responds, a logic 1 signal is put out at output LIMACT. To prevent repeated changing-over in the limit ranges, the output is given a fixed hysteresis of 0.5 %.

“P“ Step P_STEP

Summary

The element P_STEP (**“P“ STEP** function) is used within a sequential control to give the commands step by step for running (ON program) or stopping (OFF program) of a functional group. One element corresponds to one step. The same elements will be used for the ON program as well as for the OFF program.



PC element P_STEP

Call P_STEP

Terminal Description

No	Name	Type	Description
1	STDCONI	IG	**ST**andar**D CON**nection **I**n. Connection from a P_STEP or a GRC element.
2	TMON	IT	**MON**itoring **T**ime.
3	TDLY	IT	**D**e**L**a**Y T**ime.
4	STEPNUM	II	**STEP NUM**ber
5	ALTNUM	II	**ALT**ernative step **NUM**ber.
6	PRCCRIT	IB	**PR**o**C**eeding **CRIT**eria.
7	OVRCRIT	IB	**OV**e**R**running **CRIT**eria.
8	ALTCRIT	IB	**ALT**ernative proceeding **CRIT**eria.
10	STDCONO	OG	**ST**andar**D CON**nection **O**ut. Connection to the next P_STEP or a GRC element.
11	CRITCLL	OB	**CRIT**eria **C**a**LL** for indication of the criteria not being fulfilled. (This output is active too when input RELCMD of the element GRC is: RELCMD=0.)
12	CMD	OB	Step **C**o**M**man**D**.

NOTE

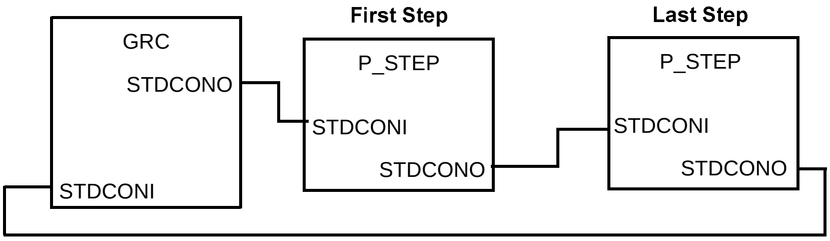
If TMON=0 the corresponding step will not be monitored.

The first step of the ON program must have assigned step number 1.

Within a program, there must be a consecutive numbering of the steps.

Connections of the element GRC and P_STEPs

Data transfer between the element GRC and the elements P_STEP will be done via the connection STDCONI to STDCONO. All steps and the GRC have to be included in this chain.



Function

Step number

Each step in a chain of steps must be identified with an unique ascending step number. The first step of the ON program must have assigned step number 1. On the basis of the value in the STDCONI-STDCONO telegram given by the GRC element, the step decides if he will be active or if the information in the telegram will be forwarded to the next step in the chain.

Step command output

One cycle after actuation of the step, the step command at output CMD will be carried out if a logic 1 signal is present at input RELCMD of the element GRC. (Release for commands)

Continuous running of the program by proceeding criteria

The process criterion (proceeding condition) will be connected to input PRCCRIT. When the proceeding condition is given, there will be forwarded to the next step under consideration of an eventually defined delay time TDLY. (Refer to „Delay time“.)

Overrunning criteria

For synchronizing of the control with process state (search program), it will be necessary to overrun certain steps without a given command. For that purpose, input OVRCRIT will be used. A logic 1 signal at this input immediately effects forwarding to the next step without a given command. An eventually defined delay time TDLY will be ignored.

Alternative proceeding criteria

Input ALTCRIT is used to realize jump programs. A logic 1 signal at this input effects jumping to the step which is specified at input ALTNUM after an eventually defined delay time TDLY has expired.

Time monitoring

The time for executing each command can be monitored by the time which is specified at input TMON. The permissible time can be adjusted for each step. The supervision function of the time will be done by the element GRC as well as the generation of a respective annunciation of an alarm when the time has been exceeded. If the monitoring time of a step shall be put out of use, a logic 0 signal at input TMON must be present.

Delay time

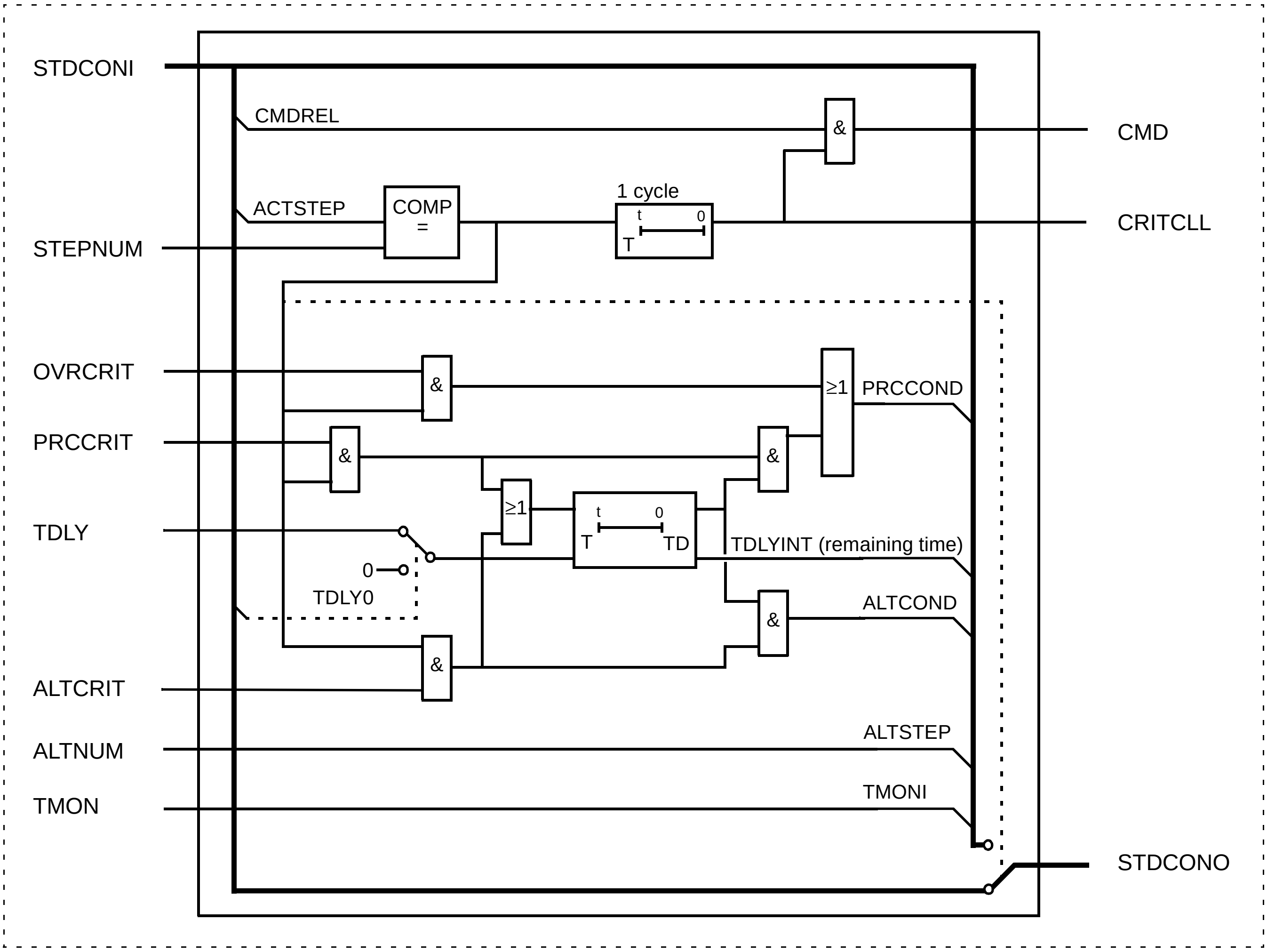
A delay time can be defined at input TDLY. With actuated step and given proceeding criteria or alternative proceeding criteria, the step will delay output of the respective signal to the element GRC according to the delay time which is specified at input TDLY. Additionally after a criteria was given, the step computes the remaining delay time and gives this message to the GRC.

Criteria call

Output CRITCLL is used to indicate criteria not yet being fulfilled. One cycle after actuation of the step, this output will carry out a logic 1signal. The output CRITCLL will be active too when input RELCMD of the element GRC is RELCMD=0.

Indication of a step

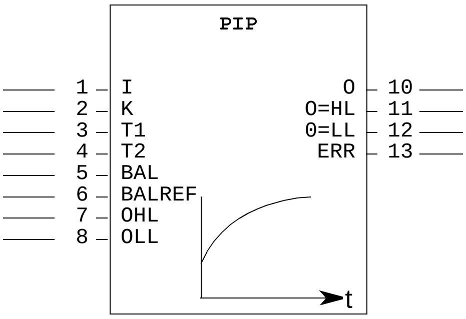
The active (set) step will be indicated at output ACTSTEP of the element GRC.



Function diagram

PIP-Function PIP

Summary



PC element PIP

PIP (**P**roportional **I**ntegrating **P**roportional function) is used to give a proportional effect and a limited integration effect. The output signal in the steady state is proportional to the input signal. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.

Call PIP

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T2T1	ITR	**T**ime **1**. Input for time constant for lead.
4	T2	ITR	**T**ime **2**. Input for time constant for lag.
5	BAL	IB	**BAL**ance. Input for activation of following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.

Function, Transfer Function

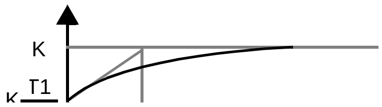
The step response in the time plane for the PIP function is:

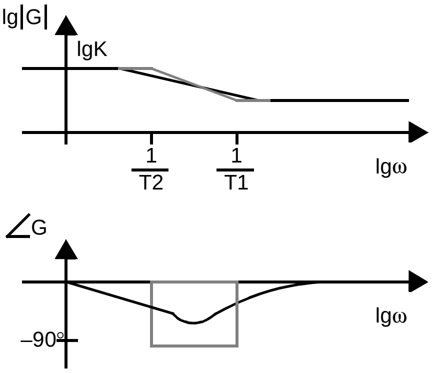
$O(t)=I(t)\ (K+k1xe^{-t/T2})$ where $k1=K\ (T1/T2-1)$

The transfer function for a PIP function is:

$G(s)=K(1+sT1)/(1+sT2)$ where $T2>T1$

This has been implemented in the PIP element as a recursive function.





Bode diagram

Gain, Time Constants and Sampling Time

Certain constants are pre-calculated to make the execution time of the element as short as possible. The result is stored internally in the element. These constants are recalculated if T1, T2 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check if T1 and $T2 \geq 2 \times TS$. If this is not the case, T1 or T2 respectively is set equal to $2 \times TS$.

Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

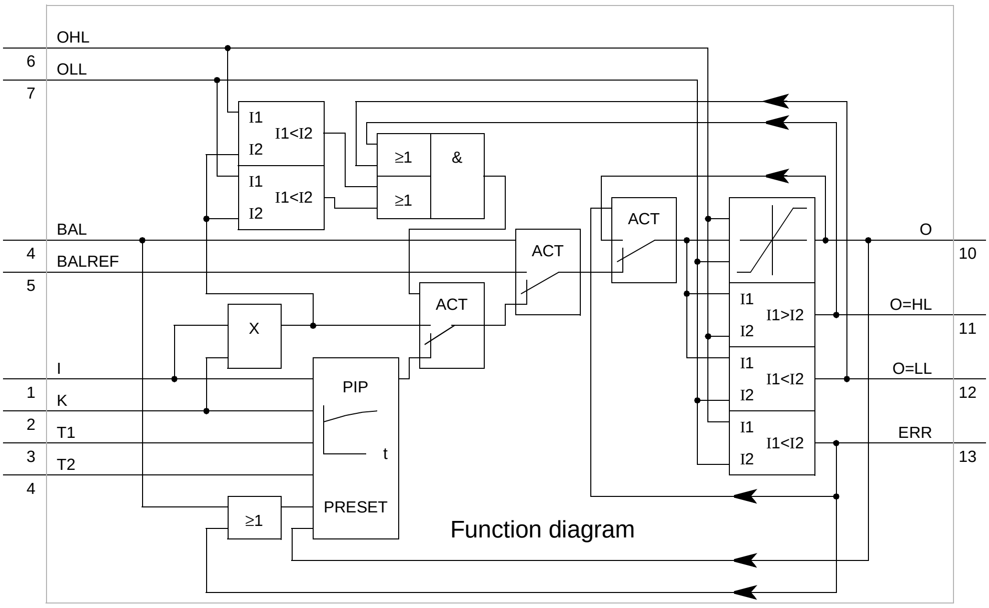
Limitation Function

The limitation functions limits the output signal to the values at the inputs OHL for the upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

When the limitation status has been detected, a check is made each time the element is executed to determine whether $K \times I(t)$ exceeds the output signal limitations. If such is the case, the limitation status remains. If not, the calculation of the output signal is performed by the algorithm in the normal way.

Normal updating of the internal status is sufficient to ensure bumpless return to the dynamic status. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1.

While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following.



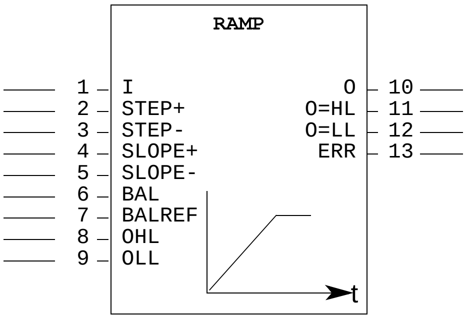
Function diagram

Ramp Generator **RAMP**

Summary

RAMP (**RAMP** generator) is used to limit the speed of change of a signal. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference.

Call RAMP



PC element RAMP

Terminal Description

No	Name	Type	Description
1	I	IR	Input. Input for the start value concerned.
2	STEP+	IR	The greatest allowed positive step change.
3	STEP–	IR	The greatest allowed negative step change.
4	SLOPE+	IR	Positive ramp for the output.
5	SLOPE–	IR	Negative ramp for the output.
6	BAL	IB	**BAL**ance. Input for activation of following.
7	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
8	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
9	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput=**H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput=**L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.

Function

The main property of a RAMP function is that the output signal follows the input signal while the input signal is not changed more than the value specified at the step inputs. If the input signal is changed more than so, the output signal is first changed by STEP+ or STEP– depending on the direction of change and then by SLOPE+ or SLOPE– per second until the values at the input and output are equal. This means also that if STEP– =STEP+=0, a pure RAMP i.e. SLOPE/sec is obtained at the output.

The greatest step change allowed at the output O is specified by the parameters STEP+ and STEP– for the respective direction of change. The ramp which the output signal is to follow if the change at the input I exceeds STEP+ or STEP– is specified by the inputs SLOPE+ and SLOPE–. All parameters are specified as absolute values with the same units as input I.

The values of the parameters are stored internally in the element. New values are only entered under stationary conditions, i.e. when I(t)=O(t).

Certain constants are pre-calculated to make the execution time of the element as short as possible. The results are stored internally in the element. These constants are recalculated if SLOPE+ or SLOPE– are changed by more than 1/128 part of their previous values or if the sampling time TS is changed.

The second figure below shows the relationship between input, output and internal auxiliary variables when the element functions normally.

Calculation of Output Signal

When calculating output signal O, three cases must be distinguished:

- I(t)=O(t-TS) results in O(t)=I(t)
- I(t)>O(t-TS) results in calculation of the value of the output in accordance with:
  O(t)=min(I(t) and (Step++VPOS(t–TS)+SLOPE+ × TS), i.e.the least of the values of I(t) and
  (Step++VPOS (t–TS)+SLOPE– × TS)
- I(t)<O(t–TS) results in calculation of the value of the output in accordance with: O(t)=max(I(t),
  VNEG (t-TS)–STEP—SLOPE– × TS

VPOS and VNEG are auxiliary variables with positive and negative steps respectively. These are calculated in all cases according to:

- VPOS(t)=min(O(t), VPOS(t–TS)+SLOPE+ × TS)
- VNEG(t)=max(O(t), VNEG(t–TS)–SLOPE– × TS)

If a new step with the same direction of change appears at the input before the internal auxiliary signal has been updated the step part of the output signal is reduced. A step with the opposite derivative, however, takes full effect. This is because the auxiliary variable for the opposite direction always has been updated to O(t).

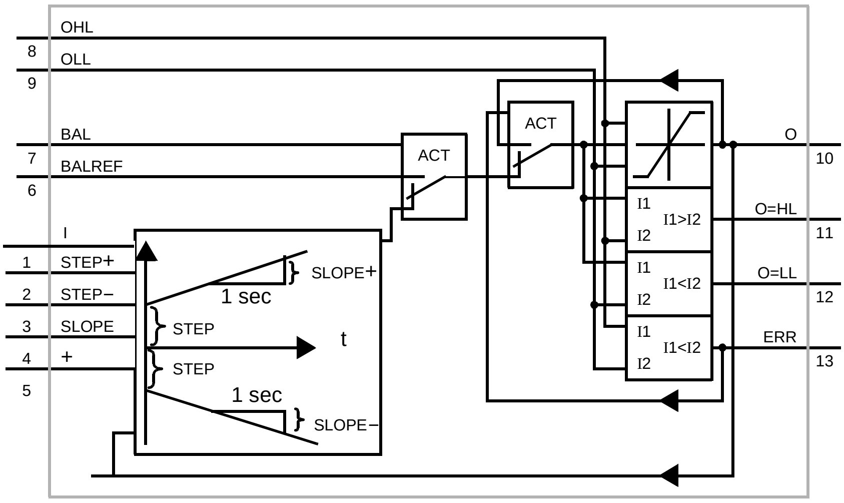
Following

If BAL is set to 1, the filter immediately goes into following and the output O is set to the value of the input BALREF.
If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. During following VPOS(t)=VNEG(t)=O(t)=BALREF(t). Return to normal function is done as if a unit step had occurred on the input.

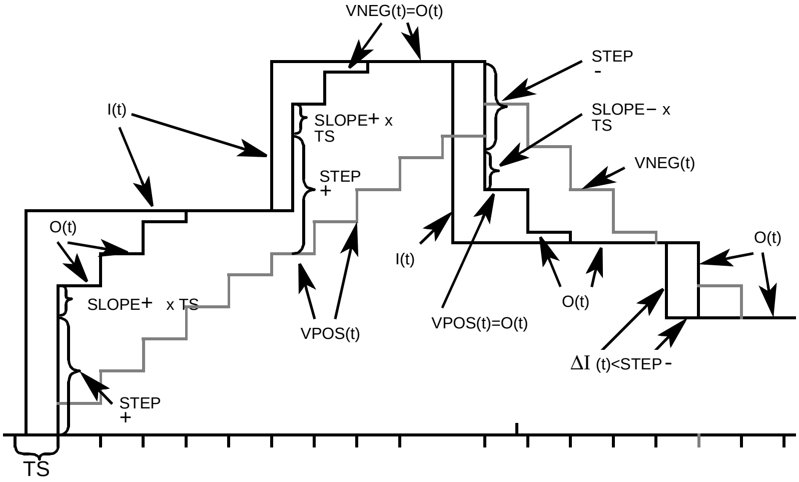
Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1. In the limiting state VPOS(t), VNEG(t) and O(t) are set to the limit value concerned.

The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred.



Function diagram

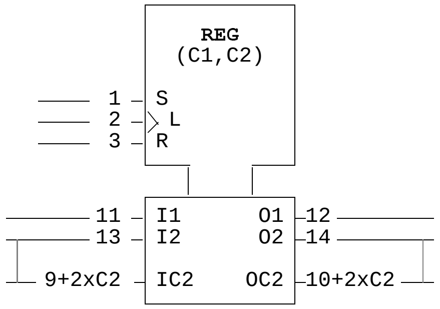


Relation between input, output and auxiliary variables under normal conditions

Register REG

Summary

Register REG (**REG**ister) is used as a memory element with an optional number of positions. The value of the data can be integer, real number, Boolean or time.



PC element REG

Call REG (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of places	1..35

Terminal Description

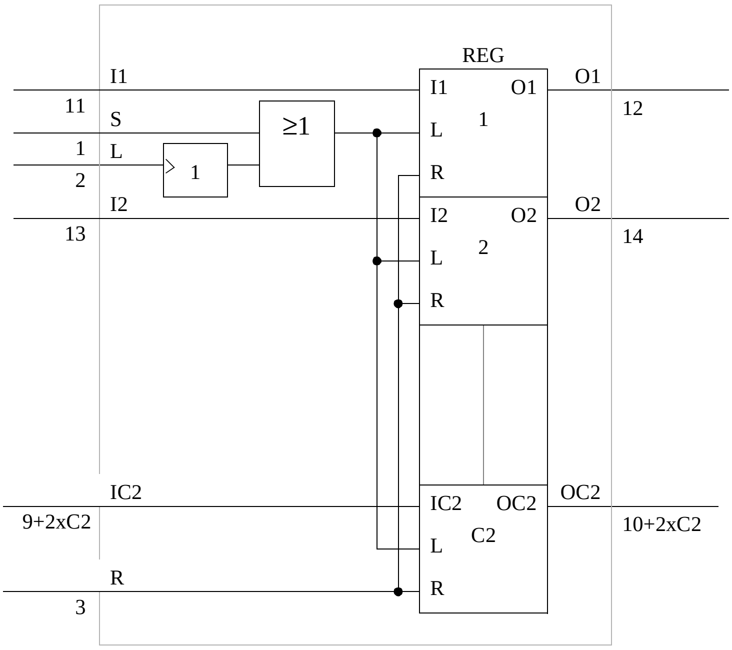
No	Name	Type	Description
1	S	IB	Set. Input for loading new data, each time the element is executed. When S is set to 0, the last data loaded remains.
2	L	IB	**L**oad. Dynamic input for loading data to the register. Loading is performed when L goes to 1.
3	R	IB	**R**eset. Input for clearing the register. R prevents all further entry while it is 1.
11	I1	I*C1*	**I**nput **1**. Input data to position 1.
12	O1	O*C1*	**O**utput **1**. Output data from position 1.
13	I2	I*C1*	**I**nput **2**. Input data to position 2.
14	O2	O*C1*	**O**utput **1**. Output data from position 2.
: 9+2xC2	I*C2*	I*C1*	**I**nput ***C2***. Input data to position C2.
10+2xC2	O*C2*	O*C1*	**O**utput ***C2***. Output data from position C2.

Function

When input L becomes 1, the register is loaded with data from inputs I1 … I*C2*. Data previously in the register is replaced. If input S is set, loading is performed as above each time the register is executed. When S is reset after having been set, the data most recently loaded remains until the element is executed again, with input S, L or R set. Input S overrides L so that when input S is set, L has no effect.

Clearing

When input R is set, the register is cleared and all further entry is prevented. R overrides both S and L.

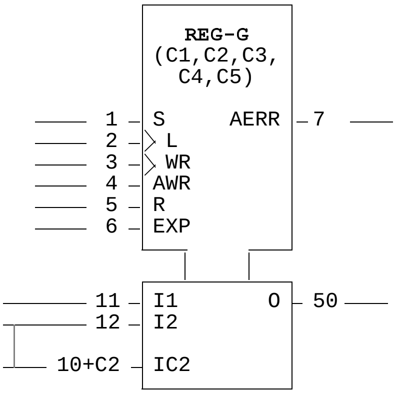


Function diagram

Group Data Register REG-G

Summary

REG-G (**REG**ister **- Gr**oup data) is used to assemble individual variables to a single variable of group type. The data type can be integer, real number, Boolean or time.



PC element REG-G

Call REG-G (C1, C2, C3, C4, C5)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	0..30
C3	Selection of expander input EXP	0..1
C4	Number of places in group data for expander input EXP	0..254
C5	Number of places in group data for the output O.	0..255
C2+C3 x C4=C5		

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading new data each time the element is executed. The latest loaded data remains when S is reset.
2	L	IB	**L**oad. Dynamic input for loading data to the register. Loading is performed when L goes to 1.
3	WR	IB	**WR**ite. Dynamic input for changing of data at the place specified by AWR.
4	AWR	II	**A**ddress **WR**ite. Input for address to the data to be changed according to WR.
5	R	IB	**R**eset. Input for clearing the register. R prevents all further entry while at 1.
6	EXP	IG*C4C1*	**EXP**ander input. Input for group data to be linked with data at inputs I1 .. I*C2*.

Terminal Description

No	Name	Type	Description
7	AERR	OB	**A**ddress **ERR**or. Output which is set (to 1) if the address at AWR is greater than the number of inputs or is negative.
11	I1	I*C1*	**I**nput **1**. Input data to position 1.
12	I2	I*C1*	**I**nput **2**. Input data to position 2.
: 10+C2	I*C2*	I*C1*	**I**nput ***C2***. Input data to position C2.
50	O	OG*C5C1*	**O**utput. Output for group data.

Function

REG-G combines individual variables of the same data type to a single group variable. The element has an expander input EXP for a group variable which can be used when more than 30 variables are to be linked together. Input S overrides L so that when input S is set, L has no effect.

Normal Assembly

When input S is set, data is continuously assembled at the group variable of the output. The group variable of the output consists firstly of group data from the input EXP and then of the values from the inputs I1 … I*C2*. The element acts as a latch when the input S is reset, the latest data assembled then remaining at the output.

Loading

If input L goes from 0 to 1, an assembly is performed to output O during this program cycle in the same way as during normal assembly. If S or R is set, L has no effect.

Changing of Data at an Optional Place

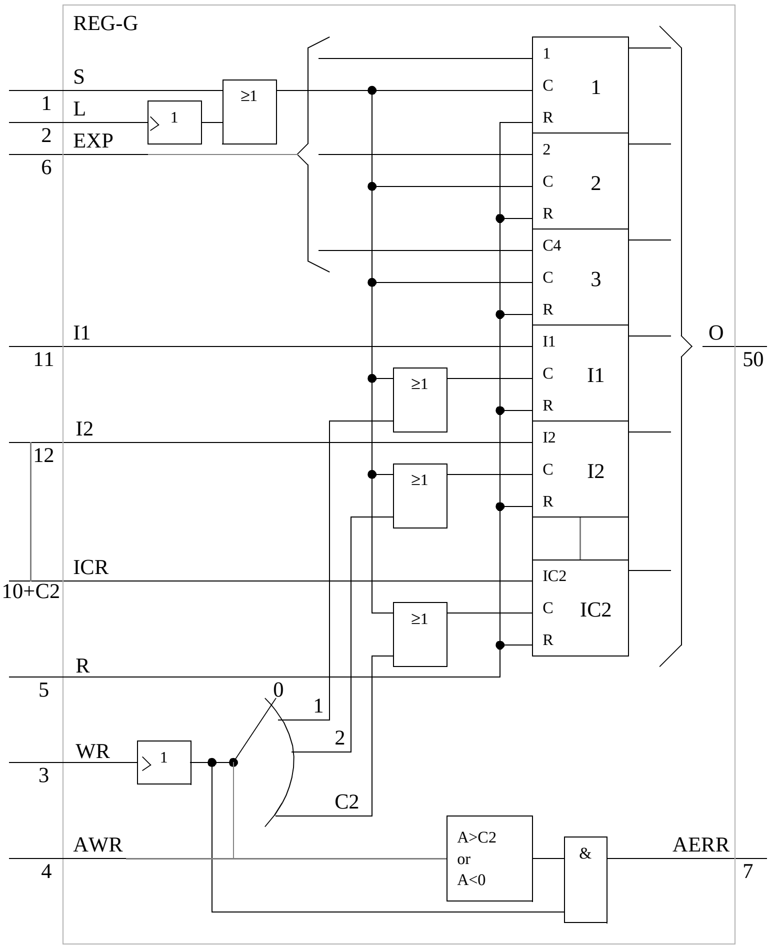
One of the ordinary data inputs I1 … I*C2* is used for changing of data at a special place in REG-G. The address (integer 1 … C2) of the place at which data is to be changed is stated at the input AWR. The new data value is entered from the input to the relevant place at the moment that the input WR goes to 1. If AWR = 0 and the input WR goes to 1, group data is read from the input EXP to their different places. Places which correspond to ordinary inputs are not affected.

Clearing

When input R is set, data at all places in the group register are cleared and all further entry is prevented. R overrides both S and L.

Supervision

The address at the input AWR is checked and if its value is greater than the number of inputs or negative, the error signal output AERR is set. When an error is detected AERR is set during one cycle if input WR is set. No place in the register is affected when an error occurs.

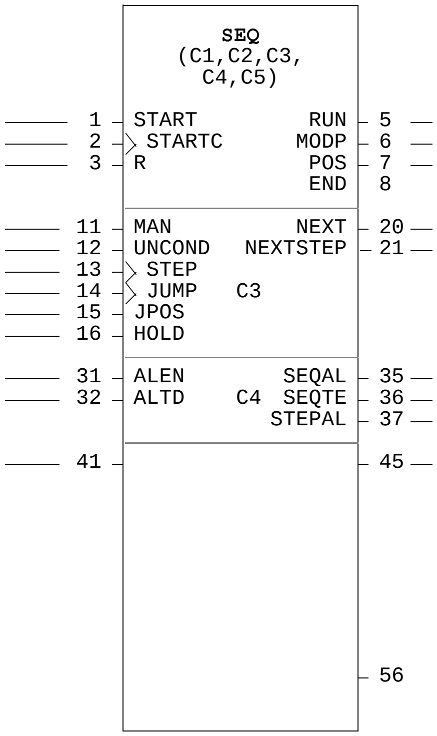


Function diagram

Sequence Header SEQ

Summary

SEQ is used to control a sequence.



PC element SEQ

Call SEQ (C1, C2, C3, C4, C5)

Parameter	Significance	Permissible values
C1	Cyclicity in ms	Allowed values are between 2ms and 20000ms in steps of 2ms
C2	Place in the cycle time table and/or scheduling strategy	1..251: place in cycle time table, cyclic activation 252: activation at initialization after power down. 254: activation at warm start initialization after power down.
C3	Selection of manual control	0..1
C4	Selection of time function	0..1
C5(1)	Selection of MMI function	Same values allowed as in AC 400 Series and MP 200/1 but in AC 110, **no effect.**

(1) Parameter C5 is allowed for compatibility reasons to the AC 400 Series or MP 200/1-system. You can exchange AMPL source code between AC 400 Series or MP 200/1 and AC 110 without problems.

Terminal Description

No	Name	Type	Description
1	START	IB	Input for start of sequence.
2	STARTC	IB	**START C**lock. Dynamic input for start of the sequence. Used when the sequence is to be run through once only.
3	R	IB	**R**eset. Clearing of the elements of the sequence.
5	RUN	OB	Indicates that the sequence is in operation.
6	MODP	OB	**MOD**ify **P**ermission. Specifies if the sequence may be modified from the engineering station.
7	POS	OI	**POS**ition. Output which specifies the step number of the active step.
8	END	OB	Output which is active when the last step of the sequence is active.
11	MAN	IB	**MAN**ual. Input which is to be set for manual stepping of the sequence.
12	UNCOND	IB	**UNCOND**itional. Input which permits stepping even when the condition for the next step is not satisfied.
13	STEP	IB	Input for step change.
14	JUMP	IB	Input for jump to step specified by JPOS.
15	JPOS	II	**J**ump **POS**ition. Input which specifies the last part of the record designation for the step to which the jump is to be made.
16	HOLD	IB	Input which is used to stop the execution.
20	NEXT	OB	Output which is set when any condition for step change or jump is satisfied or when UNCOND is set.
21	NEXTSTEP	OI	Step to which stepping in manual mode is performed. Is connected to the corresponding input of the data base element when MMC function is used.
31	ALEN	IB	**AL**arm **EN**able. Input for start and stop of time measurement.
32	ALTD	IT	**AL**arm **T**ime **D**elay. Input for time after which SEQAL is to be set.
35	SEQAL	OB	**SEQ**uence **AL**arm. Output which is set when the time specified on ALTD has elapsed.
36	SEQTE	OT	**SEQ**uence **T**ime **E**lapsed. Output for elapsed time.
37	STEPAL	OB	**STEP AL**arm. Output for summary signal which indicates that the time set in an active subordinate step has elapsed.

Function

The sequence header SEQ is a superior execution controlling element for a number of sequence steps. How often the sequence is to be executed is specified with the call parameter C1 and the place in the cycle time table on which the order of execution of the execution units with the same cycle time is based is specified with the call parameter C2. The basic function can be extended with the call parameters C3, C4 and C5. Manual control of the sequence can be obtained by setting C3 to 1 at call. A time function can be obtained by setting C4 to 1.

Start of Sequence

With normal execution, the sequence can be started with a pulse at the input START. The pulse is registered in an internal memory in the sequence header. If a fixed 1 is connected to START, the sequence can be connected for automatic restart after resetting. If the sequence is to be run through only once, it must be started with STARTC. STARTC is a dynamic input and the internal memory of the sequence header is only affected when the input status is changed from 0 to 1. After a start, the output RUN is set and the condition input for the first step is checked. If the condition is satisfied, the first step and its elements are executed.

Clearing

If the input R is set, the sequence is executed in the reset mode. This means that all outputs on the elements within all of the steps of the sequence are given initial values, which in most cases is the 0-value of the data. Reading to the data base is done with the data value 0 for all data types. For the steps, reset execution is performed only during the cycle when R goes to 1. For the sequence header, reset execution is performed during each cycle while R is set to 1. Input R overrides the inputs START, STARTC and HOLD.

RUN

The output RUN is set when the sequence is under execution, i.e. when started with START or STARTC. RUN is cleared by R, HOLD and blocking functions introduced from service aids.

MODP

MODP is always true.

POS and END

The output POS specifies, with an integer (I), the last part of the record designation of the sequence step which is active. If POS is 0, the sequence header is active. The output END is set when the last step in the sequence is active.

Manual Execution

For manual stepping of a sequence, the input MAN must be set to 1. The sequence is started in the same way as with automatic stepping by setting START or STARTC to 1. When the condition for step change, or a jump condition, is satisfied, the output NEXT is set, and the step change can be performed with a pulse at the input STEP. Jumps have priority to ordinary step change.If stepping is required without the condition for the next step being satisfied, the input UNCOND must be set to 1 when STEP is set to 1.

Jump in Sequence

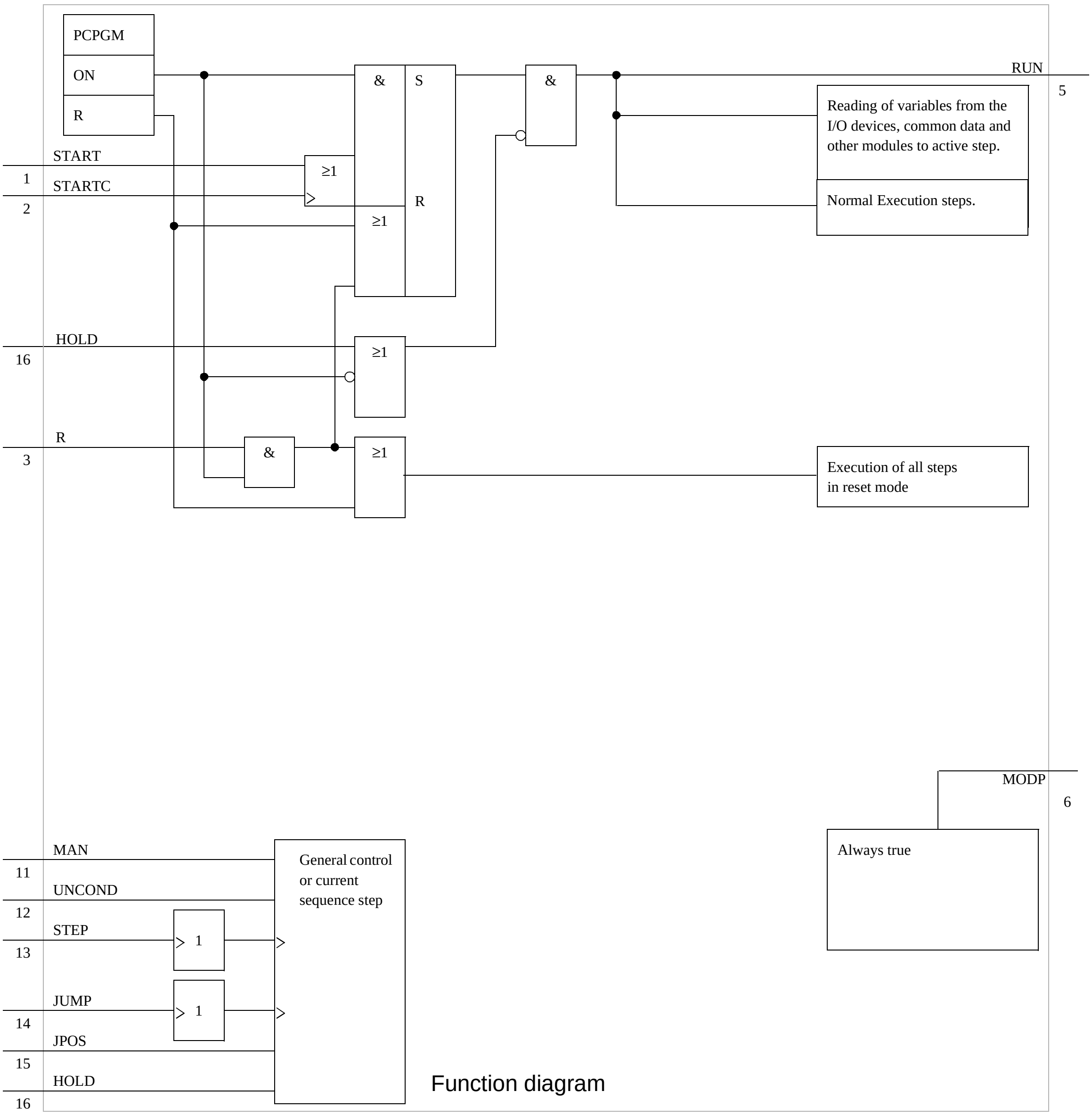
A jump can be obtained by setting the input JUMP. The jump is made to the step which is specified at the input JPOS. The step is specified by the last part of its record designation. The jump is performed irrespective of the status at the inputs MAN, STEP and UNCOND. If the MMC function is used and a jump is to be initiated by the operator via the dialogue, the operation mode must be MANM. A jump is not possible if HOLD or R is set.

HOLD

When HOLD is set, execution of active sequence steps is stopped. The calculated data remain until the next time the sequence step is executed. The execution is stopped and the output RUN is reset while HOLD is set to 1. The RUN output at the step header remains set. HOLD also stops the timers in the sequence header and the step headers.

Time Functions

If the call parameter C4 is given as 1, a time function associated with the sequence header is obtained. A common signal STEPAL is obtained which indicates any time function which has elapsed in any of the steps. The switch on of the timer in the sequence header is delayed with accumulating time measurement controlled by the input ALEN. If ALEN is reset, the timer is stopped but its contents remain and when ALEN is set, the time measurement restarts from the value that applied when ALEN was reset. The setting of HOLD has the same effect on the timer as resetting ALEN. The timer can only be started when the sequence is active. When the time specified at the input ALTD has elapsed, the output SEQAL is set. When ALTD is set equal to 0, SEQAL is set to 1 irrespective of the state at input ALEN. The output SEQTE gives the time which has elapsed with the timer active. The timer is cleared when the complete sequence is cleared with the input R. A timer corresponding to that in the sequence header can be selected for the sequence steps. The output STEPAL, which specifies if the time set in the timer for the active step has elapsed, is located in the sequence header. STEPAL is set to 1 if the delay specified for the step concerned has elapsed.

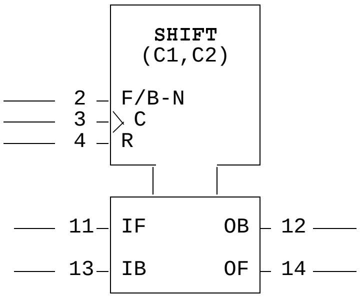

PCPGM
ON
R
START
1
STARTC
2
HOLD
16
R
3
&
S
R
≥1
≥1
&
≥1
&
≥1
RUN
5
Reading of variables from the I/O devices, common data and other modules to active step.
Normal Execution steps.
Execution of all steps in reset mode
MODP
6
Always true
MAN
11
UNCOND
12
STEP
13
1
JUMP
14
1
JPOS
15
HOLD
16
General control or current sequence step

Function diagram

Shift Register SHIFT

Summary

Shift register SHIFT is used as a shift register of up to 250 positions. Data can be integers, real numbers, Boolean or time.



PC element SHIFT

Call SHIFT (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of places	2..250

Terminal Description

No	Name	Type	Description
2	F/B-N	IB	**F**orward/**B**ackward-**N**. Input for selection of shift direction. Shifting is forward when F/B-N=1 and backward when F/B-N=0.
3	C	IB	**C**ontrol. Dynamic input for shifting.
4	R	IB	**R**eset. Input for clearing the shift register. All further shifting is prevented when R=1.
11	IF	I*C1*	**I**nput **F**orward. Input data with forward shift.
12	OB	O*C1*	**O**utput **B**ackward. Output data with backward shift.
13	IB	I*C1*	**I**nput **B**ackward. Input data with backward shift.
14	OF	O*C1*	Output Forward. Output data with forward shift.

Function, Forward Shift

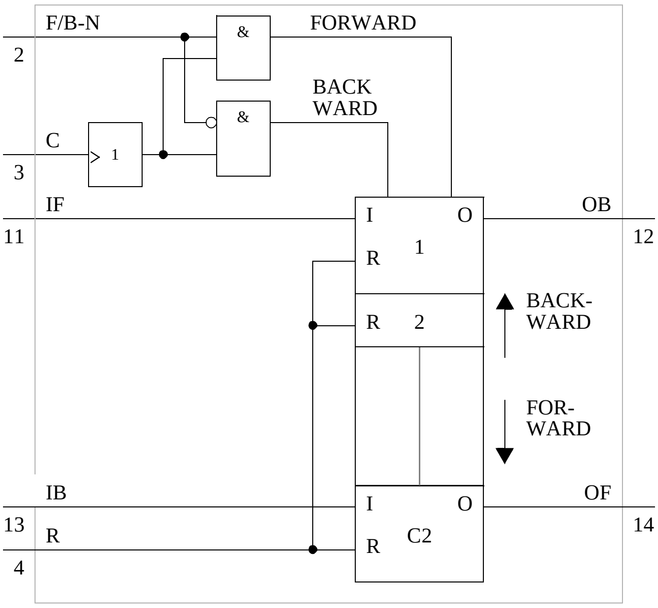
When input C goes to 1 and the input F/B-N is 1, data is shifted forwards in the register and data is read into position 1 from input IF. Data which before the shifting was located at output OF, is replaced at the shift forward by data at the position next to last.

Backward Shift

When input C goes to 1 and the input F/B-N is 0, data is shifted backwards in the register and data is read into position C2 from input IB. Data which before the shifting was located at output OB, is replaced at the shift backward by data from position 2.

Clearing

The input R clears the shift register and prevents all further reading into the shift register.



Function diagram

Slave Header SLAVEM

Summary

SLAVEM (**SLAVE M**odule) is used for execution control of a slave module.

```
SLAVEM (C1, C2)
```

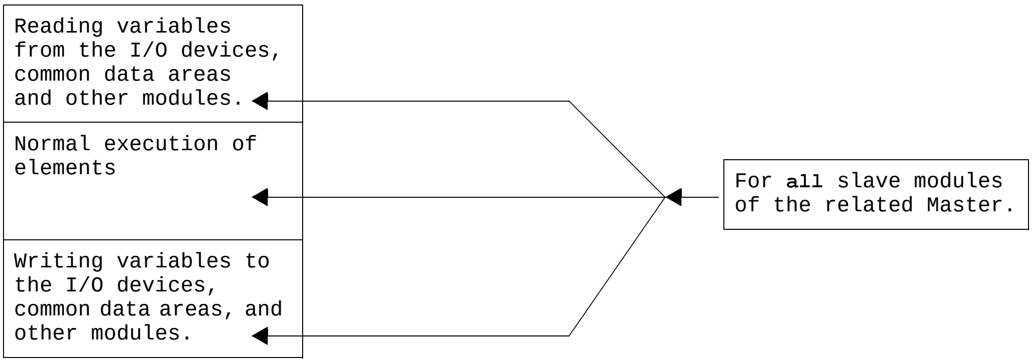
PC element SLAVEM

Call SLAVEM (C1, C2)

Parameter	Significance	Permissible values
C1	Identity of execution controlling master header	The second part of the PC record designation of the master header.
C2	Currently not used	

Function

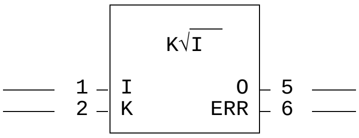
The slave header SLAVEM is an element header for a slave module. The numerical identity of the master header associated with the slave module is specified with the call parameter C1. This then becomes the second part of the PC record designation of the master as this must always be at the level directly subordinate to a program header. How often and in what order the slave module is to be executed in relation to other execution units is determined by the superior master header. The slave modules within the same master are executed in the order of the documentation. Blocking from service aids and PC programs is determined entirely by the master.


Reading variables from the I/O devices, common data areas and other modules.
Normal execution of elements
Writing variables to the I/O devices, common data areas, and other modules.
For all slave modules of the related Master.

Square Root SQRT

Summary

SQRT (**SQ**uare **R**oo**T**) is used to calculate the square root of a real number. The result can be multiplied by an optional value.



PC element SQRT

Call SQRT

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for number of which the square root is to be calculated.
2	K	IR	Input for multiplication factor K.
5	O	OR	**O**utput. Output for the product of the value at input K and the square root of the value at input I.
6	ERR	OB	**ERR**or. Output which is set if the value at input is negative.

Function

The square root of the value at input I is calculated. The result is multiplied by the value at input K. The product is stored at the output O.

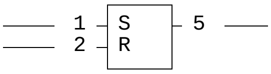
Supervision

If the value at the input I is negative, the error output ERR is set to 1. The value 0.0 is then stored at the output O.

Set/ Reset Memory SR

Summary

The memory element SR (**S**et **R**eset memory) is used as a memory for Boolean variables.



PC element SR

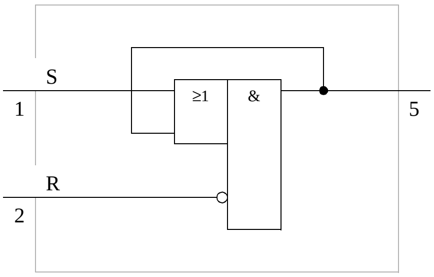
Call SR

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et input.
2	R	IB	**R**eset input which overrides the set input.
5	–	OB	Output from the memory element.

Function

The element output is set (to 1) if the set input is set at the same time that the reset input is reset (to 0). If the reset input is set, the output is unconditionally reset.

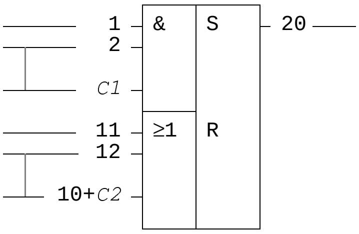


Function diagram

Memory Elements SR-AA, SR-AO, SR-OO, SR-OA

Summary

SR-AA, SR-AO, SR-OA and SR-OO (**S**et **R**eset memory - **A**=And, **O**=Or) are used as a memory for Boolean variables. The elements have different gate elements for condition generation at the set and reset inputs.



PC element SR-AO

Call SR-AA (C1, C2), SR-AO (C1, C2), SR-OA (C1, C2) SR-OO (C1, C2)

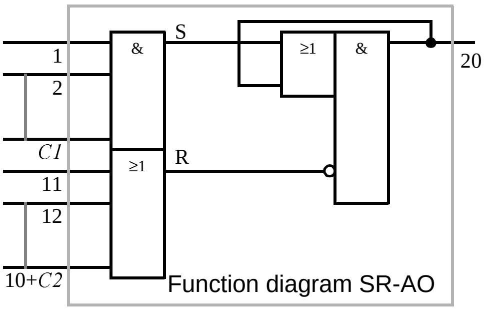
Parameter	Significance	Permissible values
C1	Number of inputs in the set gate	1..9
C2	Number of inputs in the reset gate	1..9

Terminal Description

No	Name	Type	Description
1	–	IB	Input to the set function.
2	-	IB	Input to the set function.
: *C1*	-	IB	Input to the set function.
11	-	IB	Input to the reset function.
12	-	IB	Input to the reset function.
: 10+*C2*	-	IB	Input to the reset function.
20	-	OB	Memory element output.

Function

The output is set to 1 if the conditions for the set gate (AND or OR) are satisfied at the same time that the conditions for the reset gate are not satisfied. If the conditions for the reset gate are satisfied, the output is unconditionally reset.

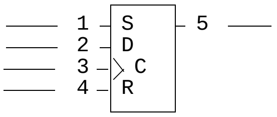


Function diagram SR-AO

Set/ Reset Memory with Data Input SR-D

Summary

The memory element SR-D (**S**et **R**eset memory - **D**ata input)



PC element SR-D

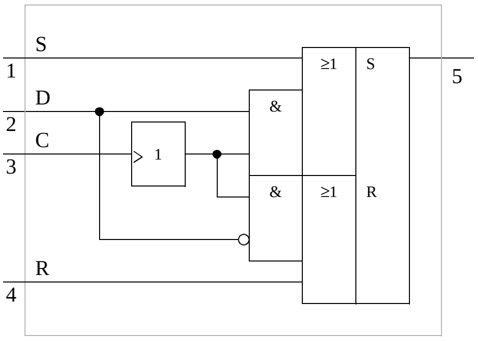
Call SR-D

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et input.
2	D	IB	**D**ata input.
3	C	IB	**C**lock. Dynamic input for entry of data from the D input.
4	R	IB	**R**eset input which overrides all other inputs.
5	-	OB	Output from the memory element.

Function

If only the S and R inputs are used, SR-D functions as an ordinary SR element. When the input R is reset and the input C goes to 1, the value at the input D is stored at the output 5. When the input R is set, the output 5 is unconditionally reset, i.e. R overrides the other inputs.

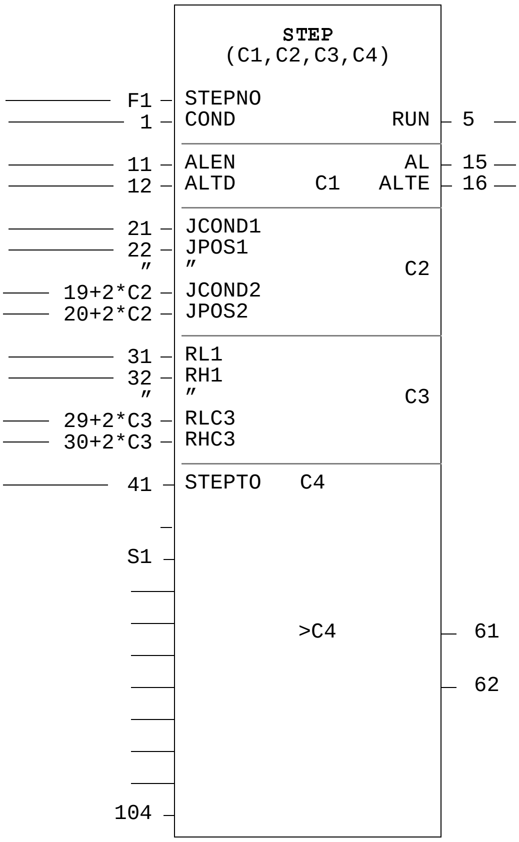


Function diagram

Step Header STEP

Summary

The step header STEP is used to partition a sequence into sequence steps.



PC Element STEP

Call STEP (C1, C2, C3, C4, C5, C6, C7, C8)

Parameter	Significance	Permissible values
C1	Selection of time function	0..1
C2§	Number of jumps	0..4
C3	Number of reset groups	0..4
C4	Selection of alternative execution order	0..1
>C4[1]	Same values allowed as in AC 400 Series or MP 200/1 but in AC 110-system **no effect.**	

(1) Parameters >C4 is allowed for compatibility reasons to the AC 400 Series or MP 200/1-system. So you can exchange AMPL source code between AC 400 Series or MP 200/1 and AC 110 without problems.

Terminal Description

No	Name	Type	Description
F1	STEPNO	II	Parameter input for **STEP N**umber. F=0 for automatic step numbering according to the last number of the item designation. F=1..999 for manual step numbering.
1	COND	IB	**COND**ition. Condition input for stepping.
5	RUN	OB	Indicates that the step is active.
11	ALEN	IB	**AL**arm **EN**able. Input for start and stop of time measurement.
12	ALTD	IT	**AL**arm **T**ime **D**elay. Input for alarm time.
15	AL	OB	**AL**arm. Output for alarm. Set (to 1) when the time specified at the input ALTD has elapsed.
16	ALTE	OT	**AL**arm **T**ime **E**lapsed. Output for elapsed time.
21	JCOND1	IB	**J**ump **COND**ition **1**. Input for jump to step specified by JPOS1.
22	JPOS1	II	**J**ump **POS**ition **1**. Specifies which step the jump is made to if JCOND1 is set.
23	JCOND2	IB	**J**ump **COND**ition **2**. Input for jump to step specified by JPOS2.
24	JPOS2	II	**J**ump **POS**ition **2**. Specifies which step the jump is made to if JCOND2 is set.
: 19+2xC2	JCOND*C2*	IB	**J**ump **COND**ition ***C2***. Input for jump to step specified by JPOS*C2*.
20+2xC2	JPOS*C2*	II	**J**ump **POS**ition ***C2***. Specifies which step the jump is made to if JCOND*C2* is set.
31	RL1	II	**R**eset **L**ow limit **1**. Input for the number of the first step in the first group of steps to be cleared.
32	RH1	II	**R**eset **H**igh limit **1**. Input for the number of the last step in the first group of steps to be cleared.
33	RL2	II	**R**eset **L**ow limit **2**. Input for the number of the first step in the second group of steps to be cleared.
34	RH2	II	**R**eset **H**igh limit **2**. Input for the number of the last step in the second group of steps to be cleared.
: 29+2xC3	RL*C3*	II	**R**eset **L**ow limit ***C3***. Input for the number of the first step in the last group of steps to be cleared.
30+2xC3	RH*C3*	II	**R**eset **H**igh limit ***C3***. Input for the number of the last step in the last group of steps to be cleared.
41	STEPTO	II	**STEP T**O. This input modifies the execution order of the sequence. The input determines which step should be next step in the sequence. To be used in convergence of a sequence (see the manual ”MasterPiece - PC programming”).

Function

The step header STEP is a supervisory element header for partitioning a sequence into sequence steps. Elements, function modules and blocks can be directly subordinate to a step header. How often and in which order the step is to be executed in relation to other execution units is determined in the supervisory sequence header.

The element has a basic function which may be extended by setting one or more of the call parameters C1, C4 to a value different from 0. A time function can be obtained with the call parameter C1. Jump and reset functions can be obtained with C2 and C3. With C4 an alternative execution order is obtained.

Step Number

If parameter input STEPNO has the default value 0, step numbering is performed automatically according to the last number of the STEP element's item designation. If you wish to have other step numbers, number 1 … 999 can be given explicitly at the STEPNO parameter input. It is the user's responsibility to give each step an unique step number.

Basic Version

The basic version of STEP includes only the condition input COND for stepping and the output RUN which indicates that the step is active.

Normal Execution

A step change takes place if the sequence is executing normally in accordance with the conditions in the sequence header and the COND input of the following step is set to 1. A step may also become active through a jump from the sequence header.
A step may become active without COND being set to 1. This occurs when UNCONDitional on the sequence header is used. With the step change, the RUN output of the active step is reset, after which the elements in this step are executed once in the normal way. The RUN output of the new active step is then set and the elements in this step are then executed. If the input HOLD on the supervisory sequence header is set to 1, execution of the active step is stopped and the RUN output remains set.

Time Function

If the call parameter C1 is specified as 1, a timer with accumulating time measurement, controlled by the input ALEN, is obtained. If ALEN is reset, the timer is stopped but the contents remain and when ALEN is set, the time measurement continues from the value which applied when ALEN was reset. Setting HOLD in the supervisory sequence header gives the same result on the time measurement circuit as the resetting of ALEN. The timer can only be started when the step is active. Elapsed time is specified at the output ALTE. When the time specified at the input ALTD has elapsed, the output AL is set. The timer is reset when the step is cleared by another step (see below) or by the setting of the R input in the supervisory sequence header.

Jump

When any of the condition inputs JCOND1 … JCOND*C2* are set, a jump is performed to the step specified at the corresponding input of JPOS1 … JPOS*C2*. If several condition inputs are set at the same time, a jump is made to the step specified by the input with the lowest sequential number. If the conditions for step change are satisfied at the same time as those for a jump, the jump is performed. No jumps can be performed when the input HOLD on the supervisory sequence header is set. If the input MAN on the sequence header is set, a jump is not made until the input STEP is set.

When executing the step, the elements included are executed first, after which the jump conditions are checked. If both the conditions for jump and those for step change are satisfied, the jump is made. If JPOS=0 for the jump input concerned, a jump is made to the supervisory sequence header. If the step specified by the JPOS input does not exist, the sequence stops.

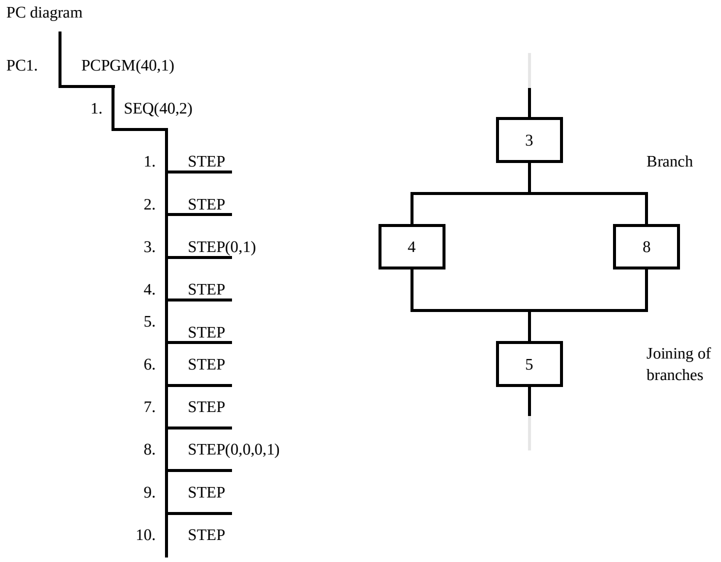
Clearing of Step

When a sequence step has been active, the data calculated remains, provided that the RUN output is not included as a condition for those data. Calculated data can be cleared by a later sequence step. One or more steps in succession can be cleared simultaneously. The step from which clearing is to be performed is specified with an integer (I) on one of the inputs RL1 … RL*C3*. The step to be cleared is specified with an integer (I) on the corresponding input of RH1 … RH*C3*. Clearing is performed when leaving the step with a normal step change, or with a step change, controlled by a jump condition. If RL=0, the complete sequence is cleared, which corresponds to a reset on the sequence header. In this case RH has no effect. If the RL value does not exist, there is no reset. If RH does not exist or has a value which is smaller than RL, the reset is performed at the end of the sequence.

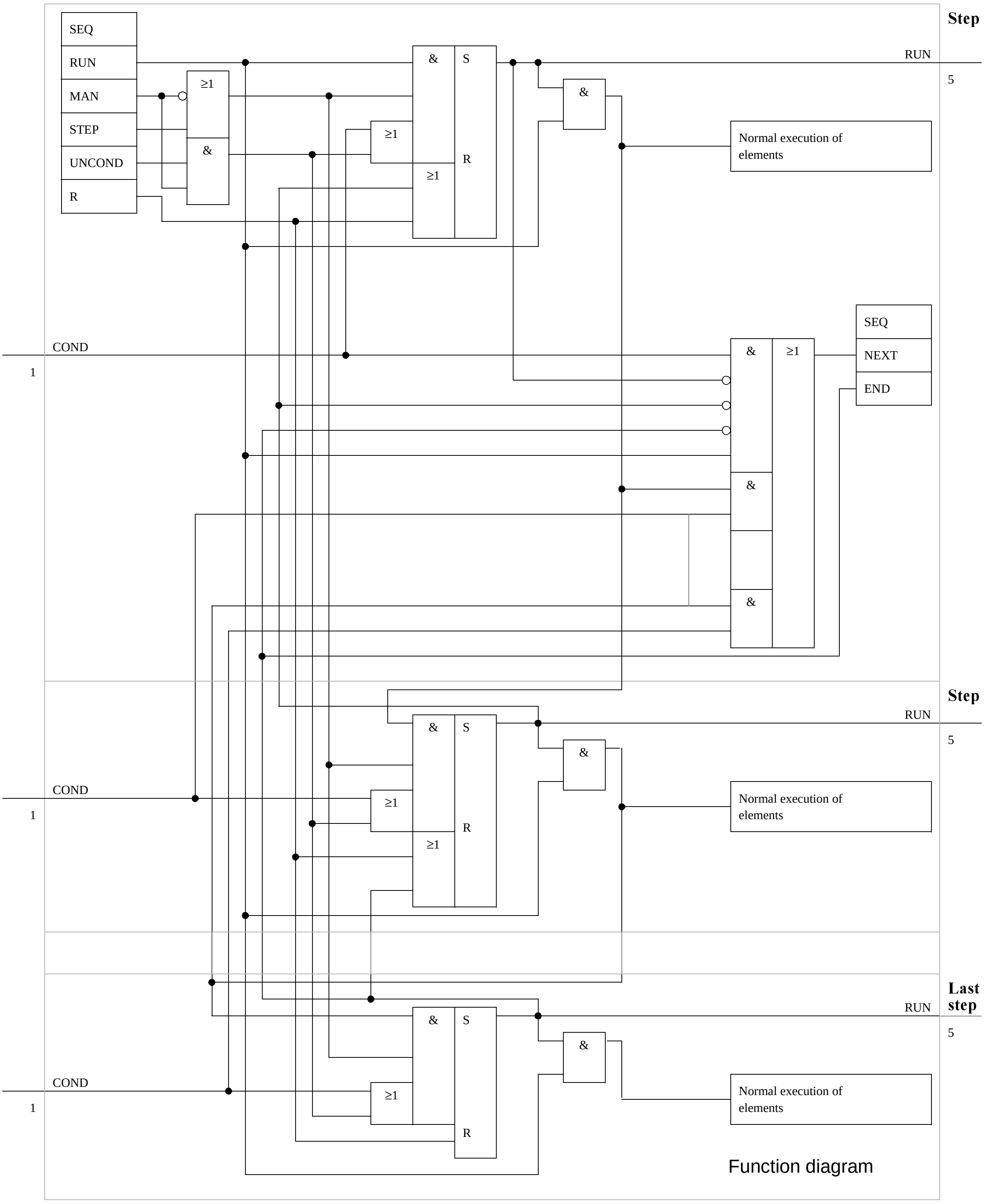
Alternative Execution Order when Joining Branched Sequences

Normally the steps are executed in the order they are build according to the PC diagram. When a sequence is branched, the jump function is used. However, when joining several sequence branches, it is better to use the alternative execution order function.

Input STEPTO then determines which step should be next step in the sequence. This step will be next step in the sequence irrespective of where it is situated in the PC diagram. Several sequence branches can have the same step determined as next step. Thus it is possible to join several sequence branches to a common step irrespective of where the steps are situated in the PC diagram. If STEPTO=0 the execution is changed to a supervisory sequence head, and if STEPTO gives a nonexisting step, the sequence is interrupted. Example: In the branched sequence in figure 116, the left branch is build as a straight, normal sequence. Branching is performed with jumps (C2=1 and JPOS1=8 in step 3) and the joining of the branches is executed with alternative execution order (C4=1 and STEPTO=5 in step 8). Thus step 5 will follow step 8 when executing the sequence. The step following step 8 in the PC diagram (in this case step 9) will not affect the execution of step 8 in any way.



Example of branched sequence with joining of branches

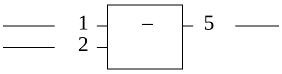


Function diagram

Subtractor SUB

Summary

SUB is used for subtraction of two integers or real numbers.



PC element SUB

Call **SUB (C1**

Parameter	**Significance**	**Permissible values**
C1	Data type	I, IL, R

Terminal Description

No	**Name**	**Type**	**Description**
1	–	I*C1*	Input for minuend.
2	–	I*C1*	Input for subtrahend.
20	–	*OC1*	Output for difference.

Function

The value at input 2 is subtracted from that at input 1 and the result is stored at output 20.

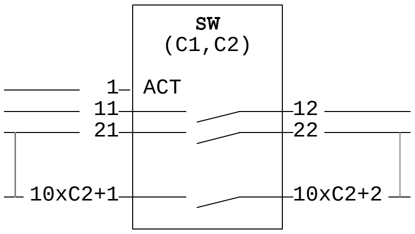
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Switch SW

Summary

SW (**SW**itch) is used as a connection element for data and has an optional number of channels with closing function. The data type can be integer, real number, Boolean or time.



PC element SW

Call SW (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of channels	1..9

Terminal Description

No	Name	Type	Description
1	ACT	IB	**ACT**ivate. Input for activation of the switch. When the input is set to 1 the switch is activated.
11	-	I*C1*	Input to channel 1 which is connected to the output for channel 1 when the switch is activated.
12	-	O*C1*	Output from channel 1.
21	-	I*C1*	Input to channel 2 which is connected to the output for channel 2 when the switch is activated.
22	-	O*C1*	Output from channel 2.
: 10xC2+1	-	I*C1*	Input to channel C2 which is connected to the output for channel C2 when the switch is activated.
10xC2+2	-	O*C1*	Output from channel C2.

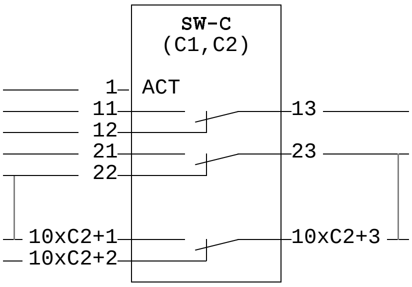
Function

When the control input ACT is 0, the output data is 0 according to the data type. When ACT is set, data comes from the inputs 11 … 10 × C2+1.

Changeover Switch SW–C

Summary

SW-C (**SW**itch **-** **C**hangeover) is used as a connection element for data and has an optional number of channels with switching function. The data type can be integer, real number, Boolean or time.



PC element SW-C

Call SW-C (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of channels	1..9

Terminal Description

No	Name	Type	Description
1	ACT	IB	**ACT**ivate. Input for activation of the switch. When the input is set to 1 the switch is activated.
11	–	I*C1*	Input for channel 1 which is connected to the output for channel 1 when the switch is activated.
12	–	I*C1*	Input to channel 1 which is connected to the output for channel 1 when the switch is not activated.
13	–	I*C1*	Output from channel 1.
21	–	I*C1*	Input to channel 2 which is connected to the output for channel 2 when the switch is not activated.
23		O*C1*	Output from channel 2.
: 10xC2+1	–	I*C1*	Input to channel C2 which is connected to the output for channel C2 when the switch is activated.
10xC2+2	–	I*C1*	Input to channel C2 which is connected to the output for channel C2 when the switch is not activated.
10xC2+3	–	O*C1*	Output from channel C2.

Function

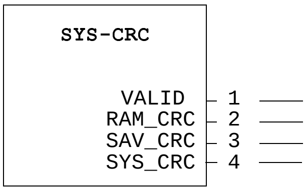
When the control input ACT is 0, the data from the inputs 12 … 10xC2+2 are connected to the appropriate outputs. When ACT is set, data comes from the inputs 11 … 10 × C2+1.

Element for Calculation of Checksums SYS-CRC

Summary

SYS-CRC (SYStem CRC) gets the values of the actual calculated and the stored Checksums.

The checksum of the System SW, the checksum of the application which runs in RAM and the checksum of the saved application in the PROM are provided.



PC Element SYS-CRC

Call SYS-CRC

Terminal Description

No	Name	Type	Description
1	VALID	OB	**VALID**. Output that is 0 as long as calculation of CRCs not finished yet.
2	RAM_CRC	OI	**RAM CRC**. Checksum of application which currently runs in RAM.
3	SAV_CRC	OI	**SAV**ed **CRC**. Checksum of the saved application in PROM.
4	SYS_CRC	OI	**SYS**tem **CRC**. Checksum of the System SW in PROM.

Function

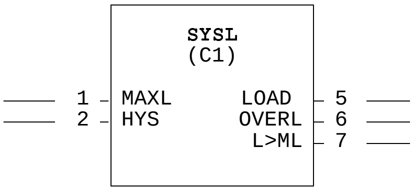
Actual CRCs are output at the terminals of the PC element.The values of the CRCs are only valid if the terminal VALID is 1.

During (re)calculation of CRCs VALID is set to 0. This may happen immediate after start up or after a new “Save and Load” command.

Element for Calculation of System Load SYSL

Summary

SYSL (SYStem Load) calculates the system load. If the system is overloaded this is indicated. Inputs and outputs for handling of maximum load may be selected with a call parameter.



PC Element SYSL

Call SYSL (C1)

Parameter	Significance	Permissible values
C1	Determines whether inputs and outputs for the maximum load is to be included in the element.	0..1

Terminal Description

No	Name	Type	Description
1(1)	MAXL	IR	**MAX**imum **L**oad. Input for maximum system load.
2(1)	HYS	IS	**HYS**teresis. Input for hysteresis for maximum system load.
5	LOAD	OR	Output for system **LOAD** in percent.
6	OVERL	OB	**OVERL**oad. Output that is set to 1 when the system is fully loaded.
7(1)	L>ML	OB	**L**oad**>M**aximum **L**oad. Input that is set to 1 when the load exceeds the value specified at the input MAXL.

(1) These connections are only included if the call parameter C1=1.

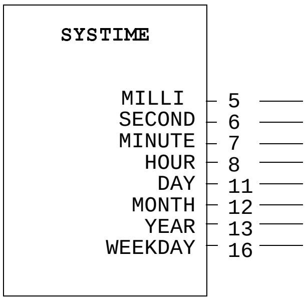
Function

If the system is fully loaded the output OVERL is set to 1. The system load in percent (%) is given at the output LOAD. If the call parameter C1 has been set to 1 the element will have inputs MAXL and HYS, for maximum load and hysteresis respectively. When the load MAXL is exceeded the output L>ML is set to one. L>ML is reset when the load lowers to below MAXL-HYS. If MAXL is higher than 100 the value 100 is used. If HYS is higher than MAXL the MAXL is used. If MAXL or HYS is negative the value 0 is used.

Time and Date Element SYSTIME

Summary

SYSTIME is used to get actual time and date



PC Element SYSTIME

Call SYSTIME

Terminal Description

No	Name	Type	Description
5	MILLI	OI	Output for millisecond (0 - 999)
6	SECOND	OI	Output for second (0 - 59)
7	MINUTE	OI	Output for minute (0 - 59)
8	HOUR	OI	Output for hour (0 - 23)
11	DAY	OI	Output for day of month (1 - 31)
12	MONTH	OI	Output for month (1 -12)
13	YEAR	OI	Output for year (>=1990)
16	WEEKDAY	OI	Output for day of week (1 - 7):Monday=1 … Sunday=7

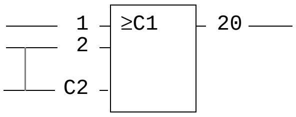
Function

Actual system time and date are output at the terminals of the PC element.

Threshold Function THRESH-L

Summary

THRESH-L (**THRESH**old **- L**ogic) is used to determine when more than, or equal to, a given number of Boolean signals are set (to 1).



PC element THRESH-L

Call THRESH-L (C1, C2)

Parameter	Significance	Permissible values
C1	Threshold value	1..C2
C2	Number of inputs	1..19

Terminal Description

No	Name	Type	Description
1	–	IB	Input for the first Boolean value.
2	–	IB	Input for the second Boolean value.
: C2	–	IB	Input for the last Boolean value.
20	–	OB	Output which is set to 1 if C1 or more of the inputs 1 … C2 are set.

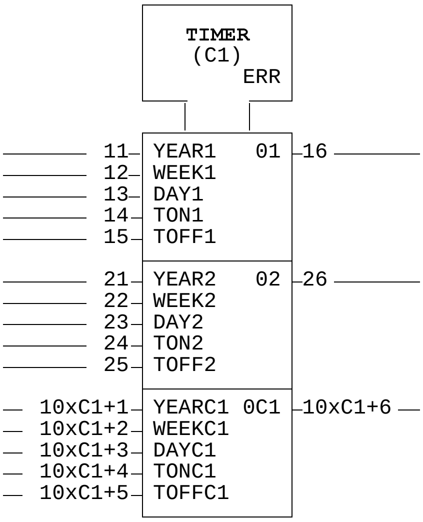
Function

The TRESH-L element is used to determine if more than, or equal to, a specified number of Boolean values in a group are set to 1. If so, the output (20) is set.

Time Selector TIMER

Summary

TIMER is used to generate a pulse with a duration determined by two times in the same (24 hour) day.



PC element TIMER

Call TIMER (C1)

Parameter	Significance	Permissible values
C1	Number of times	1..9

Terminal Description

No	Name	Type	Description
5	ERR	OB	**ERR**or. Error signal which is set to one when any of the inputs have values outside the permitted time range.
11	YEAR1	II	Input which specifies which year the pulse is to be given from the first time element. The value 0 means a pulse is to be given each year.
12	WEEK1	II	Input which specifies which week the pulse is to be given from the first time element. The value 0 means a pulse is to be given each week.
13	DAY1	II	Input which specifies which day of the week the pulse is to be given from the first time element. The value 0 means a pulse is to be given each day.
14	TON1	IT	**T**ime **ON 1**. Input which specifies at which time the pulse from the first time element is to begin.
15	TOFF1	IT	**T**ime **OFF 1**. Input which specifies at which time the pulse from the first time element is to end.
16	O1	OB	**O**utput **1**. Output at which a pulse appears between the times specified at the inputs TON1 and TOFF1.
21	YEAR2	II	Input which specifies which year the pulse is to be given from the second time element. The value 0 means a pulse is to be given each year.

Terminal Description

No	Name	Type	Description
22	WEEK2	II	Input which specifies which week the pulse is to be given from the second time element. The value 0 means a pulse is to be given each week.
23	DAY2	II	Input which specifies which day of the week the pulse is to be given from the second time element. The value 0 means a pulse is to be given each day.
24	TON2	IT	**T**ime **ON 2**. Input which specifies at which time the pulse from the second time element is to begin.
25	TOFF2	IT	**T**ime **OFF 2**. Input which specifies at which time the pulse from the second time element is to end.
26	O2	OB	**O**utput **2**. Output at which a pulse appears between the times specified at the inputs TON2 and TOFF2.
: 10xC1+1	YEAR*C1*	II	Input which specifies which year the pulse is to be given from the last time element. The value 0 means a pulse is to be given each year.
10xC1+2	WEEK*C1*	II	Input which specifies which week the pulse is to be given from the last time element. The value 0 means a pulse is to be given each week.
10xC1+3	DAY*C1*	II	Input which specifies which day of the week the pulse is to be given from the last time element. The value 0 means a pulse is to be given each day.
10xC1+4	TON*C1*	IT	**T**ime **ON** *C1*. Input which specifies at which time the pulse from the last time element is to begin.
10xC1+5	TOFF*C1*	IT	**T**ime **OFF** *C1*. Input which specifies at which time the pulse from the last time element is to end.
10xC1+6	O*C1*	OB	**O**utput ***C1***. Output at which a pulse appears between the times specified at the inputs TON*C1* and TOFF*C1*.

Function

TIMER is used to produce a pulse between two times within one (24 hours) day. The day is specified by year, week number and day number within the week. The duration of the pulse is determined by the inputs TON and TOFF. The generated pulse can be used to start required functions.

YEAR Input

The year during which the pulse is to be obtained is specified at the inputs YEAR1 … YEAR*C1*. If the year is given as 0, a pulse is given each year. In addition to 0, years 1980 to 2099 can be given.

WEEK Input

The week during which the pulse is to be obtained is specified at the inputs WEEK1 … WEEK*C1*. If the week is given as 0, a pulse is given each week. In addition to 0, week numbers 1 to 53 can be given. If week 53 is specified in a year that does not have a week 53, no pulse is produced.

DAY Input

The day of the week during which the pulse is to be obtained is specified at the inputs DAY1 … DAY*C1*. If the day is given as 0, a pulse is given each day of the week. In addition to 0, day numbers 1 to 7 can be given. Day 1 refers to Monday and day 7 is Sunday.

Rules for Week and Day Numbering

Weeks and days are numbered in accordance with the ISO standard. Monday is considered to be the first day of the week. The first week which contains at least 4 days is considered to be the first week of the year. When days are referred to by year-week-day, the change of year is on week 1, day 1.

Example: 1 Jan 1995 is given as 1994 52 6

TON and TOFF Inputs

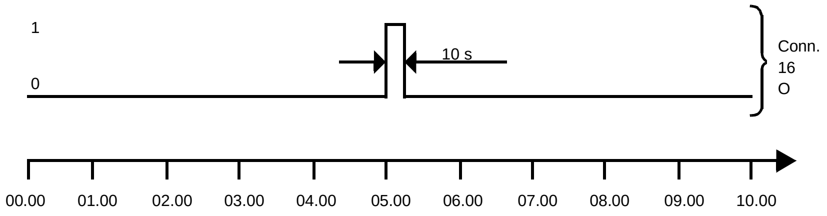
At which time a pulse is to begin or end respectively is given at the inputs TON and TOFF. TON and TOFF must be within the same 24 hour period, i.e. TON<TOFF. Time specifications can vary from 00:00:00.000 to 23:59:59.999. For repetition within a 24 hour period, e.g. each hour, external elements are required.

Examples of Selection of Day

YEAR	WEEK	DAY	Significance
0	0	0	Each day.
1980 .. 2099	0	0	Each day during a certain year.
1980 .. 2099	1 .. 53	0	Each day during a certain week during a certain year.
1980 .. 2099	0	1 .. 7	A certain day each week during a certain year.
0	1 .. 53	0	Each day during a certain week every year.
0	0	1 .. 7	A certain day of each week, every year.
0	1 .. 53	1 .. 7	A certain day of each year.
1980 .. 2099	1 .. 53	1 .. 7	A particular day.

Supervision

If any input has a value outside the permitted range, the error signal ERR is set (to 1).

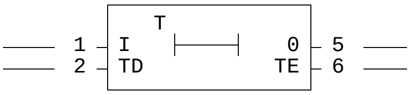


Time pulse diagram

Time Delay On/ Time Delay Off TON/ TOFF

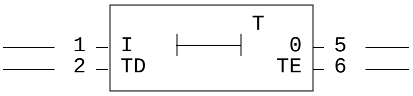
Summary

Time delay on TON (**T**imer **ON**-delay) and time delay off TOFF (**T**imer **OFF**-delay) of Boolean variables for use in connection with combined expressions.



PC element TON

Call TON TOFF



PC element TOFF

Terminal Description

No	Name	Type	Description
1	I	IB	**I**nput for start of time delay.
2	TD	IT	**T**ime **D**elay. Input for preset time. Max 23h 59m 59.999s.
5	O	OB	**O**utput. Output which is set (TON) or reset (TOFF) when the preset time has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output which specifies how long I has been set (TON) or reset (TOFF). When the preset time (TD) has elapsed, TE stops.

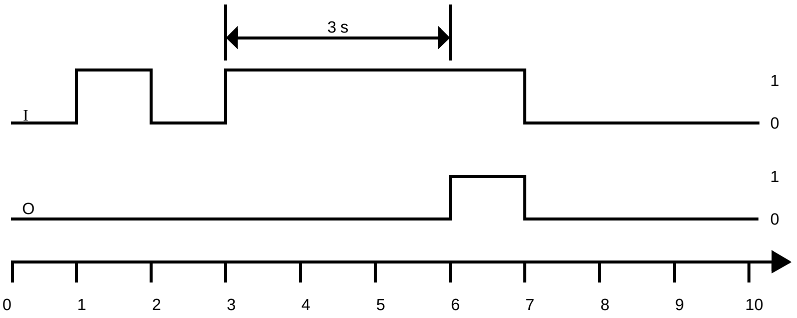
Function

TON

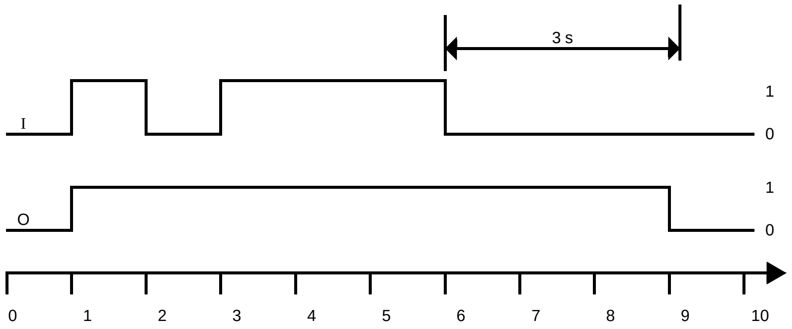
The input variable to the input I is obtained delayed at output O when the input variable changes from 0 to 1. The output signal returns immediately the input variable changes from 1 to 0.

TOFF

The input variable to the input I is obtained delayed at output O when the input variable changes from 1 to 0. The output signal is set immediately the input variable changes from 0 to 1.



Example of time diagram for TON with preset time 3 s

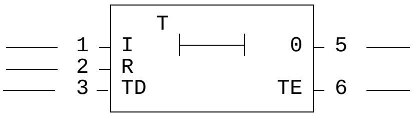


Example of time diagram for TOFF with preset time 3 s

Time Delay On with Retention TON-RET

Summary

TON-RET (**T**ime delay **ON - RET**entive) switch-on delay with accumulating time measurement of Boolean variables for use in connection with combined expressions. TON-RET is used for e.g. operational time monitoring.



PC element TON-RET

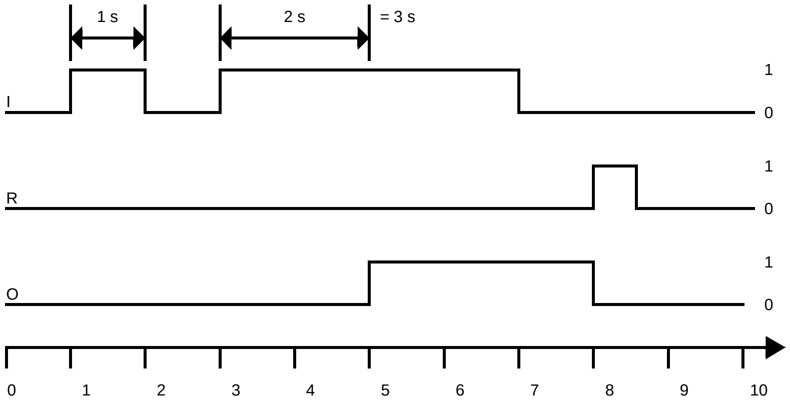
Call TON-RET

Terminal Description

No	Name	Type	Description
1	I	IB	**I**nput for start of time delay when the input changes from 0 to 1. The time circuit maintains its status even when the input returns to 0. **O**utput which is set when the preset time has elapsed.
2	R	IB	**R**eset. Input which resets the timer. R must be reset (to 0) for the timer to function.
3	TD	IT	**T**ime **D**elay. Input for preset time, i.e. the time during which the input I must be set (to 1). Max 23h 59m 59s.
5	O	OB	**O**utput which is set when the preset time has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output which indicates the time during which I has been set to 1.

Function

The input variable to the input I is obtained delayed at the output O when the input variable changes from 0 to 1 in accordance with the time pulse diagram in figure 2. If the input variable returns to 0, the time which has elapsed remains in the timer and when the variable returns to 1, the time continues from the value which applied when the variable went to 0. The input R must be reset (to 0) when the timer functions. If R is set (to 1) both outputs O and TE are reset.

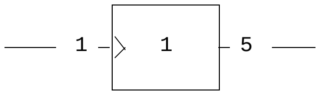


Example of time diagram with preset time 3 s

Trigger TRIGG

Summary

The triggering element TRIGG is used for, for example, reducing impulse times at the start of automatic procedures and for calculating functions.



PC element TRIGG

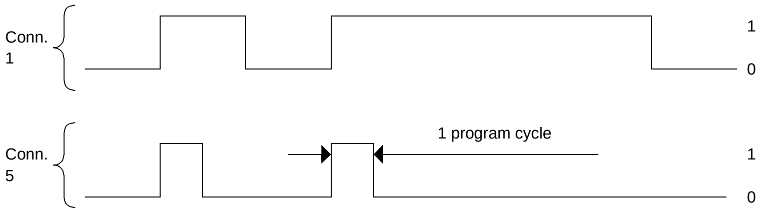
Call TRIGG

Terminal Description

No	Name	Type	Description
1	–	IB	Input for start of trigger pulse.
5	–	OB	Output for trigger pulse.

Function

When the input signal is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the value of the input signal. For the output signal to be set, the input signal must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.

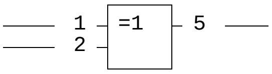


Time diagram

Exclusive Or Gate XOR

Summary

XOR (e**X**clusive **OR**) is used to generate combined expressions with Boolean variables.



PC element XOR

Call XOR

Terminal Description

No	Name	Type	Description
1	–	IB	Input
2	–	IB	Input
5	–	OB	Output

Function

The output signal from the XOR element is 1 if the input signals are different and 0 if they are equal, see the table below. By inverting the output of the XOR element, a 1 is obtained if the signals are equal.

Table Input and Output Signals

1	2	5
0	0	0
0	1	1
1	0	1
1	1	0

3BSE 000 504R301

January 1999